Delete-and-Conquer: Rateless Coding with Constrained Feedback

Morteza Hashemi*, Ari Trachtenberg*, Yuval Cassuto† *Dept. of Electrical and Computer Engineering, Boston University, USA †Dept. of Electrical Engineering, Technion, Israel Institute of Technology

A version of this work appeared as:

- M. Hashemi, A. Trachtenberg, Y. Cassuto, "Delete-And-Conquer: Rateless Coding with Constrained Feedback", 51st Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2013.

Abstract

Traditional rateless codes were designed without the use of a feedback channel, though one is available in many applications. In this work, we build upon recent interest in rateless coding with feedback to produce a novel approach, dubbed Delete-and-Conquer coding, for rateless coding with very little feedback. In our scheme, the feedback used is a measure of distance between a received word and the symbols already decoded at the receiver. This distance, in turn, permits a transmitter to deduce which symbols have been decoded and exclude them from subsequent transmissions. Our approach can be tuned to the specific transmission properties of a given feedback channel, and we empirically show that a very small amount of feedback from receiver back to the transmitter can significantly reduce coding overhead and encoding/decoding complexity. We also provide some analytically backed intuition for this improvement.

I. Introduction

Reliable communication over erasure channels has emerged as a key technology for various Internet applications (e.g., digital video broadcasting). In applications where there exists a high-throughput feedback channel, automatic repeat request (ARQ) protocols can achieve capacity over erasure-based links. However, when such feedback channels are not available, rateless codes, such as the capacity achieving Luby-Transform (LT) [1] and Raptor codes [2], can often provide reliable communication for sufficiently long block lengths.

There have been several methods proposed in the literature for exploiting a feedback channel to improve coding performance on short and intermediate block lengths, where LT codes are less efficient. The authors in [3] propose real-time oblivious erasure codes, which utilize feedback messages to send the *number* of decoded symbols to the broadcaster. Shifted LT codes [4] utilize a similar approach by *shifting* the LT codes' Robust Solution distribution [1] according to the number of decoded symbols noted through the receiver's feedback channel. The authors in [5] use more informative feedback messages to notify the transmitter of *which* input symbols have been decoded, and this information is then applied to modify the degree distribution. Typical performance criteria for such feedback-based schemes include:

- **coding overhead:** number of encoded symbols needed to decode all input symbols with a high probability;
- **computation cost:** the amount of arithmetic operations needed in the encoder and decoder;
- **utilization of feedback channel:** the amount of feedback transmitted through the back channel.

We propose a novel coding scheme, dubbed Delete-and-Conquer, that optimizes for these same metrics by using a different type of feedback message. Our feedback contains information on the *distance* between a received encoding symbol and the set of already decoded symbols at the receiver. The encoder uses this feedback to infer which symbols are known to the decoder, and those symbols are excluded from future transmissions.

For applications with constrained feedback channels, a Delete-and-Conquer receiver *opportunistically* sends distance feedback messages when certain conditions are met. In our case, feedbacks are sent for every received symbol of distance 0 or 1 from the set of decoded symbols at the receiver. This type of $0-1$ distance message is indeed a generalization of ACK-based protocols to the coded cases, as they notify the recovery of input symbols involved in a linear equation. For even more constrained feedback channels, we consider scenarios when the receiver sends a $0-1$ feedback message with a fixed probability, allowing us to trade off the forward and feedback transmissions according to the costs and constraints of both channels. We also discuss the most general form of our approach, wherein *all* forward transmissions trigger distance feedback.

It should be noted that we base our Delete-and-Conquer codes upon LT codes merely for ease of exposition; the same methodology can apply to other rateless codes already in the literature. Our use of an LT base allows Delete-and-Conquer codes to enjoy the same simple encoding and decoding procedures as LT codes, together with similar analytical guarantees, while improving overall performance with only the nominal utilization of a feedback channel. Finally, we compare our scheme in simulation to traditional LT codes, Growth codes [6], and online rateless codes proposed in [7].

In summary, the contributions of our work are:

- We propose Delete-and-Conquer rateless codes based on a distance-type feedback;
- We provide precise analyses of Delete-and-Conquer codes for very short block lengths;
- We experimentally validate the performance gains of our scheme against some existing rateless codes.

A. Organization

The rest of this paper is organized as follows. In Section II we review various related coding schemes, and specifically, in Section III, we provide some preliminaries of LT and online codes, which will be used in later comparisons. Section IV describes the intuition of our Delete-and-Conquer coding scheme adapted for constrained feedback scenarios, followed by its generalizations. Coding analyses are presented in Section V, and simulation results are provided in Section VI. We conclude with overall thoughts in Section VII.

II. Related Work

Both fixed rate low-density parity-check (LDPC) codes [8] and Turbo codes [9] are capable of correcting bit errors, as well as erasures. Byers *et al.* in [10] have presented fixed rate Tornado codes as a class of simplified capacity-achieving LDPC codes. Within the context of rateless coding, random linear codes (e.g.[11]) are well known due to their low communication overhead, but the encoding and decoding computations make them practical only for small message sizes. On the other hand, Luby Transform (LT) [1] codes and their extensions such as Raptor codes [2] are examples of rateless codes that are asymptotically optimal and also have computationally efficient encoding and decoding algorithms; unfortunately, they usually have poor performance for small block sizes [5] and various optimization methods have been proposed [12] for these cases.

In some applications, like video streaming, *intermediate symbol recovery* is also important, as it is desirable to decode some symbols before an entire frame has been received. The authors in [13] design different degree distributions for high intermediate symbol recovery rates. More recently, there have also been proposed rateless protocols that utilize side information fed back from the decoder to the encoder in order to improve the performance. In the Real Time (RT) oblivious codes [3], the encoder starts with degree one symbols and the encoded symbol degree increases based upon feedback messages containing the number of recovered symbols. Shifted LT codes (SLT) [4] use the same type of feedback information as RT codes, but instead of increasing the encoded symbols degree explicitly, SLT shift the Robust Soliton distribution according to the number of decoded symbols. The authors in [14] study the problem of minimizing the amount of feedback in broadcast scenarios by combining extreme value theory with rateless coding. As another class of feedback-based coding schemes, the authors in [7] propose online codes, in which the decoder informs the encoder about which degree it should use for next encoded symbols. Finally, the proposed scheme in [5] uses more informative feedback messages noting recovered symbols *explicitly* to the encoder, at which point the recovered symbols are excluded and the degree distribution is redesigned for subsequent encodings.

Our work differs from the cited works above in two key aspects: (1) we use a limited amount of distance-type feedback messages for rateless codes, (2) our encoder excludes the recovered symbols from the future transmissions, in addition to changing the degree distribution. As an example, though the work in [5] (described above) is similar to our approach, its goal is to design a "good" degree distribution after each deletion step, and, in addition, their scheme feeds back messages to inform the transmitter about the exact ID of recovered input symbols, which may require a significant communication overhead for the feedback channel.

III. Preliminaries

We next describe our problem and some rateless codes.

A. Setup

We assume that an encoder (broadcaster) has k input symbols to transmit to all receivers over an erasure channel, and that there exists a feedback channel through which receivers can send some information back to the encoder. In contradistinction to previous works, we do not assume that the feedback channel is high bandwidth and error-free; instead, we strive for a parsimonious use of the feedback channel. Indeed, we send back exactly one bit (plus some header information) for each of a small fraction of received symbols. For sake of clarity, we restrict our attention to the scenarios with one broadcaster and one receiver, although we do briefly describe the applicability of our codes to the cases of multiple receivers in Section IV-C.

B. LT codes

1) Encoding: Luby Transform's codes (LT) were first proposed in [1] and provide a rateless coding scheme that has low overhead. Specifically, they support full recovery with probability $1-\delta$ and computational complexity $O\left(k\ln(k/\delta)\right)$ using an expected $k+O\left(\sqrt{k}\ln^2(k/\delta)\right)$ error-free transmissions. The encoding process involves an encoder picking a coding degree d, based on the Robust Soliton distribution [1], and then adding together d distinct symbols (chosen uniformly at random) from the k input symbols to produce an encoding symbol in a given finite field. The d chosen symbols are *neighbors* of the encoded symbol, and their indices are made available to the decoder through explicit communication, e.g., by adding meta-information to the encoding symbols. In the design of Robust Soliton distribution, it is presumed that the encoder does not utilize any side information fed back from the decoder.

2) Decoding: The LT decoder (so-called Peeling decoder) uses a simple greedy algorithm, whose complexity is typically less than traditional Gaussian elimination methods. In one variant, the decoder finds all encoding symbols with degree 1, whose neighbor can be immediately recovered. These recovered input symbols are then excluded from all output symbols that have them as neighbors, reducing the number of unknowns in those encoding symbols by one. This process continues until there exists no encoding symbol with degree 1. Decoding succeeds if all input symbols are recovered; alternatively, decoding fails if, at some point, there is no output symbol with degree 1.

C. Online codes

There also exist rateless-type codes with *real-time* properties that allow intermediate knowledge of some input symbols as the decoding progresses. The authors in [6] propose Growth codes for the purposes of data collection within lossy sensor networks. Growth codes' degree increases as the coding progresses, while in the coding scheme of [7], the receiver has the ability to control the decoding progress by requesting particular encoding degrees. In this method, the average number of output symbols n required for decoding k input symbols is shown to be upper bounded by $1.236k$, referred as redundancy overhead 0.236.

IV. Delete-and-Conquer Codes

In many communication systems, the bandwidth is mainly provisioned for forward transmissions, and, therefore, it is advantageous to design feedback-based codes with a nominal utilization of the back channel. Within this context, we develop Delete-and-Conquer codes that use a very small amount of feedback, which in turn can significantly improve performance.

A. Delete-and-Conquer codes: Primitive form

There are several types of feedback messages that have been used with rateless coding in the literature:

- the receiver sends the *number* of decoded symbols to the transmitter [3, 4];
- the receiver notifies the transmitter of *which* input symbols have been recovered [5]; or
- the receiver suggests to the transmitter what kind of degrees it should use for future encodings [7].

Our approach uses feedback messages containing the *distance* of received symbols to the set of already recovered symbols at the receiver. More precisely, given a set $\mathcal{C}$ of recovered symbols at the receiver and an encoding symbol y, we are interested in the number of neighbors of y that are not present in $\mathcal{C}$.

Based on distance definition, a Delete-and-Conquer receiver transmits feedback messages only in the case of distances 0 or 1. Assuming that an encoded symbol y is received by the receiver and the set of decoded symbols

is $\mathcal{C}$, then if the distance between y and the set $\mathcal{C}$ is equal to 0 or 1, then all neighbors of y have been decoded (distance 0) or can be decoded by an exclusive or operation (distance 1). Under such cases, the receiver transmits one bit as feedback message to inform the transmitter that all neighbors of the previously received symbol are decoded and can be excluded from future transmissions. Thereafter, the transmitter deletes the neighbors of y and then rescales the primary degree distribution (e.g., Robust Soliton) to the smaller set of input symbols.

Example: Suppose that 4 input symbols x_i $(i = 1, ..., 4)$ are encoded and transmitted in the following order: $y_1 = x_1 \oplus x_2$, $y_2 = x_1 \oplus x_4$, $y_3 = x_4$, and $y_4 = x_1 \oplus x_2$. Based on distance definition, y_1 and y_2 would have distance 2, y_3 distance 1, and y_4 distance 0 (as x_1 and x_2 will be decoded after receiving y_3). The Delete-and-Conquer scheme therefore transmits a feedback message after receiving y_3 and then after y_4 signalling that x_4, x_1, x_2 (in that order) have been decoded and can be excluded from subsequent transmissions.

The benefits of Delete-and-Conquer scheme are twofold compared to the previous feedback-based methods: (1) it limits the amount of feedback messages, which are sent only in the cases of distance 0 or 1; (2) it reduces the average degree of encoding symbols by excluding the recovered symbols from future transmissions, and thus, it decreases the expected computational complexity in the encoder.

Rate-limited feedback: In the case of severe constrained feedback scenarios, we add the mechanism of *probabilistic* feedback control, in which the receiver transmits feedback only with a given probability. This probability can be determined according to feedback channel rate, condition, and the cost of back transmission. With this method, the receiver can consider the utility and costs of transmitting feedback and accordingly choose the optimal feedback probability to minimize an appropriate cost function. We briefly discuss the problem of utility-cost tradeoffs for forward and back channels in the form of two examples in Section V, and further experimentally illustrate the effect of such probabilistic feedback control in Section VI.

B. Generalization

The Delete-and-Conquer solution for constrained feedback cases can be generalized to the scenarios with a low-cost feedback channel, which provides the possibility of transmitting feedback message for each received symbol. In the generalized version of Delete-and-Conquer, each forward transmission triggers a feedback message, containing the distance information; the encoder then process the collected distance information to infer which symbols have been decoded up to the current step, and excludes those symbols from future transmissions. In order to process distance information, we propose a *distance graph* structure to find the recovered symbols.

One may argue that it is possible to simply send an ACK (without distance information), based on which the transmitter can learn about the recovered symbols, but under such circumstances, coding may not be even necessary for point-to-point communications. If the ACK channel is error-prone, then the coding can improve performance, as it potentially prevents from retransmission of already received symbols, in the cases of ACK loss. Moreover, distance-type feedback messages are applicable to the broadcast scenarios as discussed in Section IV-C, while a simple ACK method may not fit into those cases.

1) Bipartite Distance Graph: In the general form of Delete-and-Conquer scheme, the goal of the encoder is to label each input symbol as either "decoded" or "not-yet-decoded" in a manner that is consistent with the distance information, noting that there may be more than one such labeling. We therefore propose the use of a *bipartite*

(a) Before applying the Case 1 labeling

(b) After applying the Case 1 labeling

Fig. 1. **Case 1: Full-distance** (a) The encoding symbol $y_2 = x_4 + x_5 + x_6$ has full distance 3 to the current state of decoder. (b) The distance graph of (a) after one time running of Case 1 labeling.

distance graph, which is depicted in Fig. 1(a) for a simple example with two consecutive encoding symbols y_1 and y_2. In this graph, input symbols are shown on the top and the encoding symbols on the bottom; each encoding

(a) Before applying the Case 2 labeling

(b) After applying the Case 2 labeling

Fig. 2. **Case 2: Zero-distance** (a) In Case 2, there exist some symbols with distance 0. (b) The distance graph of (a) after applying Case 2 algorithm. Case 1 algorithm can be used for this graph, declaring the symbols x_1, x_2, x_5, and x_6 as decoded symbols.

symbol is connected to its neighbors through an edge with weight equal to the distance of that encoded symbol from the set of already decoded symbols. For example, y_1 is of distance 2 from the decoded set, meaning that only two of the input symbols x_1, x_2, x_3, x_4, x_5 are not known at the decoder.

Our labeling proceeds recursively, relying on two cases where the distance information fed back from the receiver to the transmitter provides unambiguous information on which symbols have been decoded at the receiver. The first case occurs when the distance of the encoded symbol is equal to its degree (number of neighbors), in which case none of the symbol's neighbors are known at the decoder (full-distance). The second case occurs when the distance of the encoded symbol is 0, implying that all of the neighbors are known at the receiver (zero-distance).

Case 1: (Full-distance) In this case, we start with an encoding symbol whose distance is equal to the coding degree, e.g., y_2 in Fig. 1(a). The labeling algorithm finds neighbors of this encoding symbol with degree more than one, such as x_4, and deletes the edge between y_2 and x_4 together with any edge connecting x_4 to other encoding symbols. Accordingly, the algorithm reduces the weight of edges from involved encoding symbols by one unit, resulting in Fig. 1(b). This algorithm continues until there is no encoding symbol whose distance is equal to its degree, whereupon all input symbols with weight zero (e.g., x_1, x_2, and x_3 in this example) can be labeled "decoded" and excluded from future encodings.

Case 2: (Zero-distance) In this case, there exists an encoding symbol with distance 0, implying that all of its neighbors have already been decoded and can be excluded from the future transmissions, as in Fig. 2(a) for encoding symbol y_3. The Case 2 labeling algorithm repeats (until exhaustion) by identifying a symbol of distance 0 and then finding its neighbors that have degree more than one, e.g., x_5, deleting all edges from the encoding symbol to these neighbors, together with edges coming out of these neighbors (resulting in Fig. 2(b), for which Case 1 now applies, allowing the encoder to infer that the symbols x_1, x_2, x_5, and x_6 are decoded and can be excluded from future transmissions).

C. Multiple receivers

Traditional rateless codes make no use of the feedback channel, making them applicable to the broadcast scenarios; on the other hand, exploiting feedback-based coding schemes under such scenarios may not be straightforward, as we assume that the transmitter broadcasts the encoded symbols to receivers, and that there exists a dedicated feedback channel between each receiver and the transmitter. In the Delete-and-Conquer scheme, excluding a *subset* of recovered symbols from the subsequent transmissions may increase the total number of transmissions, but does not impede coding progress. In particular, for the case of no exclusion, Delete-and-Conquer will reduce to the traditional LT codes. Therefore, for the multiple receivers scenario, if the set of symbols labeled decoded is not the same across the receivers, the transmitter will exclude those symbols that are decoded by all receivers. In the worst case, no symbol is dropped from the encoding set, which reduces the Delete-and-Conquer codes to the original LT codes. Alternatively, the encoder can simply take the maximum over distance values collected from different receivers, and then proceeds with the labeling algorithms based on the maximum distance.

V. Coding Analysis

In this section, we first provide some analysis results of the Delete-and-Conquer scheme, followed by an analytical illustration of the benefits of feedback for small block lengths $k = 2$ and $k = 3$. To this end, we maintain the

assumption that initially there exist k input symbols and at some point, the encoder deletes d symbols and continues with a smaller set of $k-d$ input symbols.

The Delete-and-Conquer distribution is thus given by $\omega_{k,d}(i) = \pi_{k-d}(i)$ (for $i = 1, ..., k-d$), where d is the number of deleted symbols and π_{k-d} is the Robust Soliton distribution [1] calculated over $k-d$ symbols.

We may straightforwardly adapt the results of [1] to see that the average degree of an encoding symbol under the distribution $\omega_{k,d}$ is given by $\bar{D} = O(\ln(k-d))$, and its computational complexity is $O\left((k-d)\ln\frac{k-d}{\delta}\right)$. Furthermore, an encoder that deletes d symbols out of k symbols, needs to transmit *at most* $k-d+O\left(\sqrt{k-d}\ \ln^2(\frac{k-d}{\delta})\right)$ symbols so that the decoder be able to decode all input symbols with probability at least $1-\delta$. On the other hand, from the total coding overhead viewpoint, the Delete-and-Conquer performance reduces to the standard LT codes when there is no feedback, though our simulation results show that it significantly reduces the number of transmissions needed for the receiver to decode all input symbols. For instance, if only degree 1 encoding symbols are transmitted, then the Delete-and-Conquer approach avoids the coupon collector phenomenon [15] in that it deletes decoded symbols from future encodings.

A. Small block length calculation

We next precisely analyze the performance of Delete-and-Conquer codes for very short block lengths $k = 2$ and 3. Although such small block lengths are far from practical values, they can provide some insight into Delete-and-Conquer scheme. For larger block lengths, our exact calculation of overhead in terms of degree probabilities becomes unwieldy.

1) Example 1: In the first example, we consider the block length $k = 2$, in which two input symbols x_1 and x_2 are encoded as x_1 or x_2 each with probability p, and $x_1 \oplus x_2$ with probability $1-2p$.

Lemma 1. *For the block length $k = 2$, and if the probability of degree 1 symbol is $2p$, then the Delete-and-Conquer codes provide a savings of $\frac{2p^2}{1-p}$ transmissions over LT codes for successful decoding.*

Proof: We first calculate the expected number of transmissions required by the LT codes to recover all symbols. To this end, we obtain the probability of completely decoding within $n \geq 2$ transmissions. For instance, in the case of $n = 2$, the decoder should receive one of the following sets to successfully recover x_1 and x_2:

$$\{x_1, x_2\}, \{x_1, x_1 \oplus x_2\}, \{x_2, x_1\}, \{x_2, x_1 \oplus x_2\},$$
$$\{x_1 \oplus x_2, x_1\}, \{x_1 \oplus x_2, x_2\}.$$

Accordingly, the probability of decoding within two transmissions can be calculated as $4p - 6p^2$. For a general n number of transmissions one can see that the probability of recovery within the n transmissions is:

$$P(n) = 2p^{n-1}\left(p + (1-2p)\right) + (1-2p)^{n-1}\left(p+p\right);$$

and hence, the expected total number of transmissions is:

$$\bar{n}_{LT} = \sum_{n=2}^{\infty} nP(n) = \frac{4p^2 - p + 1}{2p(1-p)}. \tag{1}$$

The situation is slightly different for Delete-and-Conquer codes, which can successfully decode two symbols within $n = 2$ transmissions under the following possibilities for the received symbols:

$$\{x_1, x_2\}, \{x_2, x_1\}, \{x_1 \oplus x_2, x_1\}, \{x_1 \oplus x_2, x_2\}.$$

And the probability of terminating after two transmissions is obtained as $4p - 4p^2$. Finally, the receiver would successfully decode x_1 and x_2 within $n \geq 3$ transmissions in the case of the following received symbols:

$$\{\overbrace{x_1 \oplus x_2, ..., x_1 \oplus x_2}^{n-1 \text{ Symbols}}, x_1\},\ \{\overbrace{x_1 \oplus x_2, ..., x_1 \oplus x_2}^{n-1 \text{ Symbols}}, x_2\}.$$

The probability of successful recovery in this case would be:

$$Q(n) = (1-2p)^{n-1}\left(p+p\right), \quad n \geq 3;$$

and therefore, the expected number of total transmissions for the Delete-and-Conquer scheme is equal to:

$$\bar{n}_{Del} = 2(4p - 4p^2) + \sum_{n=3}^{\infty} nQ(n) = \frac{4p^2 + 1}{2p}. \tag{2}$$

Using (1) and (2), the expected amount of savings $\bar{n}_{LT} - \bar{n}_{Del}$ is obtained. ■

Lemma 2. *Assuming that for the block length $k = 2$, each transmission through the forward channel has a cost C_1, while each feedback transmission has a cost C_2, then the inequality*

$$\frac{1-p}{p} \leq \frac{C_1}{C_2}$$

specifies the economical region *of transmitting feedback.*

Proof: One can see that the expected number of feedback messages sent by the Delete-and-Conquer is $2p$, as one feedback is transmitted only in the cases of received symbols $\{x_1, x_2\}$ and $\{x_2, x_1\}$ (each happens with probability p), noting that we exclude the last feedback message notifying the full recovery. The last ACK message is in fact needed by other coding schemes to stop the encoder from further transmissions. Using the Lemma 1, it would be worthwhile to send feedback messages rather than sending more encoded symbols if the following inequality holds:

$$2pC_2 \leq \frac{2p^2}{1-p} C_1 \Rightarrow \frac{1-p}{p} \leq \frac{C_1}{C_2}.$$

■

2) Example 2: For the block length $k = 3$, the authors in [12] have derived the expected number of encoding symbols required by LT codes for complete decoding. In this model, the set of received symbols at the decoder defines a state of an absorbing Markov chain, and the process (i.e., transmission of encoded symbols) ends when it arrives to the absorbing state that includes all input symbols decoded. We similarly adapt this approach to obtain the Markov chain for the Delete-and-Conquer scheme with 3 input symbols, as shown in Fig. 3. This Markov chain includes states up to the permutations of input symbols, e.g., two states $\{x_1, x_2 \oplus x_3\}$ and $\{x_2, x_1 \oplus x_3\}$ are isomorphic and it is enough to consider a single unique state for each group of isomorphic states. In this figure, darker states are irreducible by the decoder whereas other states can be immediately reduced to darker ones. By constructing the state transition matrix $\mathbf{P}$ as follows

$$\mathbf{P} = \begin{pmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$

we can compute the expected number of steps (transmissions) from the initial state to the absorbing state $\{x_1, x_2, x_3\}$. In the notation, matrix $\mathbf{Q}$ represents the transition probabilities between transient states, $\mathbf{R}$ denotes the probabilities between transient states and the absorbing state, and $\mathbf{I}$ is an identity matrix.

Lemma 3. *For the block length $k = 3$, given that p_j is the probability of transmitting an encoded symbol with degree j, the expected number of transmissions required by the Delete-and-Conquer scheme for successful decoding is:*

$$\bar{n}_{Del} = \frac{1}{p_1} + \frac{p_2}{3p_1 + 2p_2} + \frac{p_2^2}{p_1 + p_2} + \frac{3p_1 - 4p_2 + p_1p_2 + p_2^2 + 3}{3 - p_2}. \tag{3}$$

Proof: In an absorbing Markov chain with a transition matrix $\mathbf{P}$ and the *fundamental matrix* $\mathbf{N} = \mathbf{I} + \mathbf{Q} + \mathbf{Q^2} + ... = (\mathbf{I} -$ the expected number of steps (transmissions) from the initial state to the absorbing one is: $\bar{n} = \boldsymbol{\pi}_0 \mathbf{N} \mathbf{c}$, where $\boldsymbol{\pi}_0 = (1\ 0\ ...\ 0)$ is the initial probability corresponding to the state of no symbols been transmitted, and $\mathbf{c} = (1\ ..\ 1)^T$

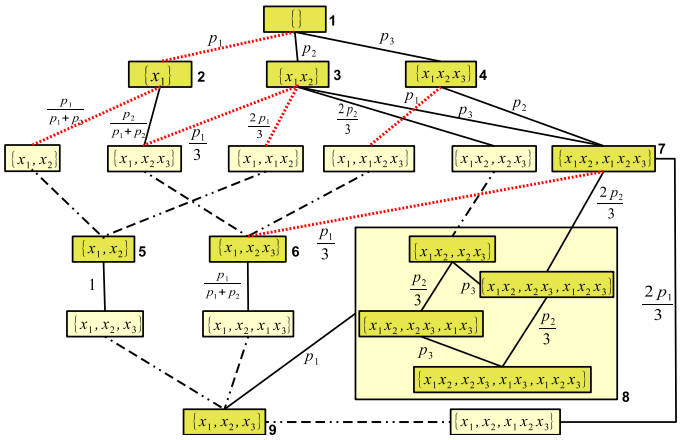


Fig. 3. State space of the Delete-and-Conquer scheme with 3 input symbols. The four states in a box are considered one state. Notation $x_i x_j$ represents the symbol $x_i \oplus x_j$, and dotted red lines represent transitions with a feedback.

[16]. From the Fig. 3, we obtain the matrix:

$$\mathbf{P} = \begin{pmatrix} 0 & p_1 & p_2 & p_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p_1' & p_2' & 0 & 0 & 0 \\ 0 & 0 & \frac{p_2}{3} & 0 & \frac{2p_1}{3} & \frac{p_1}{3} & p_3 & \frac{2p_2}{3} & 0 \\ 0 & 0 & 0 & p_3 & 0 & p_1 & p_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & p_2' & 0 & 0 & p_1' \\ 0 & 0 & 0 & 0 & 0 & \frac{p_1}{3} & \frac{p_2+3p_3}{3} & \frac{2p_2}{3} & \frac{2p_1}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-p_1 & p_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

where we assume that after each symbol deletion at the encoder, the probabilities are normalized by dividing by the sum of the remaining degrees. This leads to the lemma statement. ■

The expected number of transmissions for LT codes has been derived in [12] as follows:

$$\bar{n}_{LT} = \frac{1}{p_1} + \frac{6p_1}{p_1 - 3} + \frac{18p_1}{(3-p_2)(3-2p_1-p_2)} + \frac{9p_1}{2(p_1+p_2)(3p_1+2p_2)}. \tag{4}$$

If the encoder uses only degree 1 symbols (i.e., $p_1 = 1$), the expected number of required symbols for LT codes is $\bar{n}_{LT} = 5.5$, illustrating the effect of the coupon collector's problem; on the other hand, Delete-and-Conquer requires only $\bar{n}_{Del} = 3$ encoded symbols, which is the minimum possible number of transmissions. Using the optimizations in [12] provides $p_1 = 0.524, p_2 = 0.366, p_3 = 0.109$, resulting in a total of 4.046 transmissions, whereas Delete-and-Conquer coding with these same probabilities requires $\bar{n}_{Del} = 3.5214$ transmissions. In general, we can numerically compare (4) to (3) to see that Delete-and-Conquer improves the LT performance, and that it can decrease the total number of transmissions up to 2.4-fold.

Lemma 4. *For $k = 3$ input symbols, the expected number of feedbacks transmitted by the Delete-and-Conquer scheme before conclusion (i.e. not including the termination signal) is:*

$$\bar{f} = \frac{3p_1}{3p_1 + 2p_2} + \frac{6p_1}{3 - p_2} + \frac{p_1^2}{p_1 + p_2} - 2p_1. \tag{5}$$

Proof: In an absorbing Markov chain, the probability of ever visiting state j when starting at a transient state i is the entry h_{ij} of the matrix $\mathbf{H} = (\mathbf{N} - \mathbf{I})\mathbf{N}_{\mathbf{dg}}^{-\mathbf{1}}$, where $\mathbf{N}$ is the fundamental matrix and $\mathbf{N_{dg}}$ is the diagonal matrix with the same diagonal as $\mathbf{N}$, and $\mathbf{I}$ is an identity matrix [16]. In Fig. 3, a feedback is transmitted if transitions along the dotted-line occur, e.g. a transition from the state 1 to state 2. Accordingly, the probability of such transitions, and hence the expected number of feedback transmitted is given by:

$$\bar{f} = h_{12} + h_{12}h_{25} + h_{13}h_{35} + h_{13}h_{36} + h_{14}h_{46};$$

from which the result follows. ■

Similar to $k = 2$ symbols, we can obtain the economical region of transmitting feedback in the Delete-and-Conquer codes for the block length 3. We therefore maintain the same previous assumption of cost C_1 for forward transmission and the cost of C_2 for the back channel, and hence, it is worthwhile to send feedback if:

$$\frac{\bar{f}}{\bar{n}_{LT} - \bar{n}_{Del}} \leq \frac{C_1}{C_2},$$

where $\bar{n}_{Del}$, $\bar{n}_{LT}$, and $\bar{f}$ are calculated in (3), (4), and (5).

VI. Simulation Results

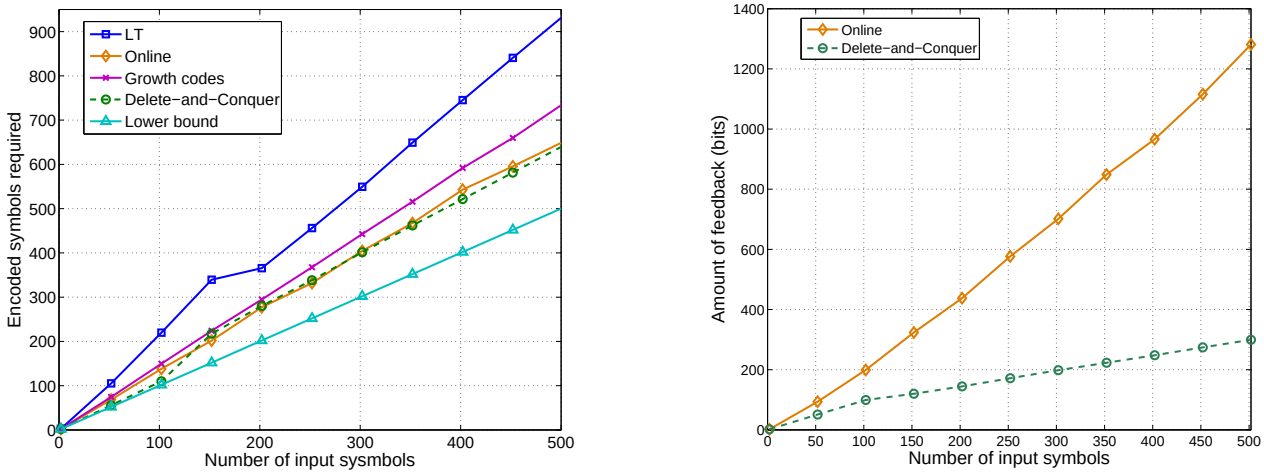
We evaluate the performance of our Delete-and-Conquer codes with respect to standard LT codes, Growth codes, and a recently proposed coding scheme [7] (referred as Online codes) through simulation. We are primarily concerned with the metrics expressed in the introduction: coding overhead, computational costs, intermediate performance, and the amount of feedback sent. The parameters c and δ in Robust Soliton distribution [1] are specified for each simulation.

a) Coding overhead: LT codes have been proved to be symptomatically optimal in terms of required encoding symbols to decode all k input symbols; however, they are also known to have poor performance for small and intermediate block lengths. Online codes are shown to have a low redundancy overhead 0.236, lower than other online schemes such as Growth codes. Our simulation results with block lengths less than $k = 500$ symbols and 1000 trials are shown in Fig. 4(a), confirming that the Delete-and-Conquer scheme improves the performance of LT codes. We also observe that Delete-and-Conquer codes have smaller overhead compared with the Growth codes and similar performance to the Online codes. On the other hand, we compare the amount (bits) of feedback sent by the Online codes and our method in Fig. 4(b), which demonstrates that the Delete-and-Conquer codes require less feedback than Online codes to achieve almost the same performance in terms of coding overhead.

b) Intermediate performance: Although LT codes are capacity-achieving, they lack real-time features; in other words, not many input symbols are decoded until the decoding process is almost complete. To investigate the progressive performance of Delete-and-Conquer codes, we run simulations with the block length $k = 128$ and 1000 trials. Our results are shown in Fig. 5, demonstrating that Growth codes can provide higher symbol recovery rate at the beginning, while Delete-and-Conquer achieves better performance when a small fraction of symbols are unrecovered (near the "knee").

c) Rate-limited feedback: As discussed in Section IV, we can add an order of flexibility to the Delete-and-Conquer receiver by probabilistic feedback control. Considering the Delete-and-Conquer scheme, every time that the receiver should transmit a feedback (i.e., a symbol with distance 0 or 1 has been received), it sends feedback only with a given probability, which can be obtained from the overall forward and feedback costs minimization. In the simulation, we increase the probability of feedback transmission, as shown in Fig. 6, and observe the effect of such increment on the coding overhead. As it is shown in Fig. 6(a), by increasing the probability of feedback the overhead $(1 + \epsilon)$ decreases, especially for short block lengths, where the original LT codes have poor performance; on the other hand, by increasing the probability, the number of feedback transmitted increases, as shown in Fig. 6(b). Therefore, based on the available rate of feedback channel and the costs of forward and back channels, the receiver can choose an optimal probability to send feedback.

d) Computational costs: Computational costs at the encoder and decoder are mainly related to the average degree of input symbols. Fig. 7 shows the average degree of input symbols for different codes compared to the Delete-and-Conquer codes. We observe that our method decreases the average degree of input symbols (hence computational complexity) by excluding the recovered symbols from the future encodings.



(a) Coding overhead

(b) Amount of feedback transmitted (bits)

Fig. 4. (a) Number of encoding symbols required by various rateless codes, and (b) feedbacks transmitted to achieve that performance ($c = 1.5$, $\delta = 0.1$).

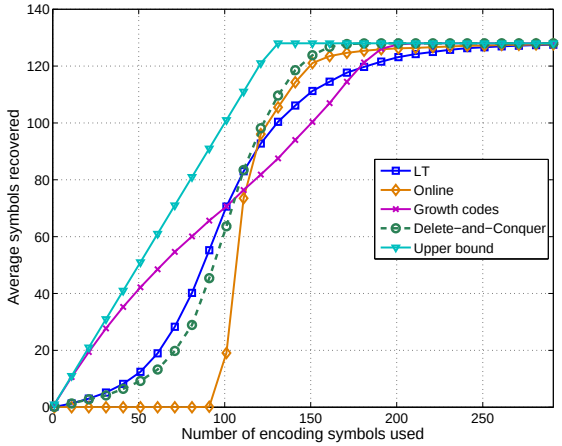
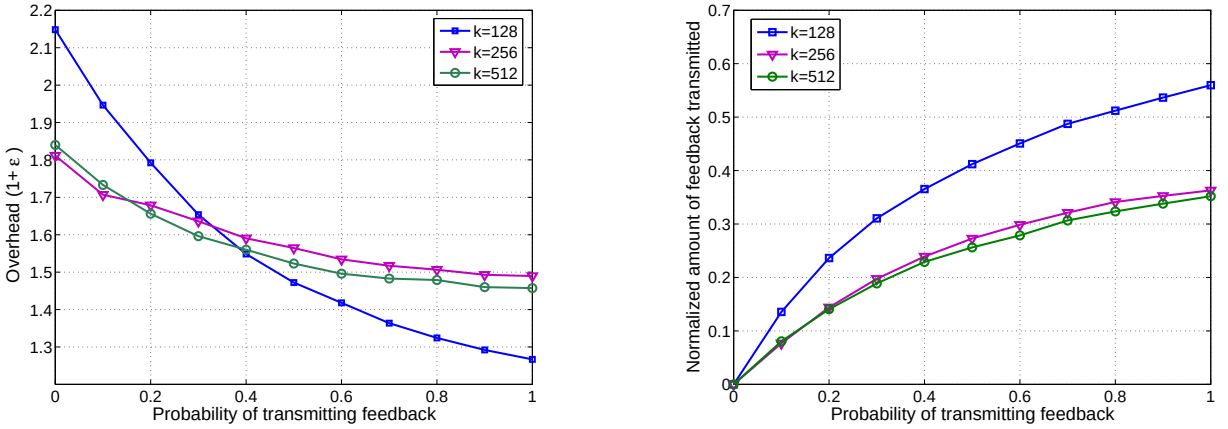


Fig. 5. *Intermediate performance* for $k = 128$ symbols ($c = 0.9$, $\delta = 0.1$).



(a) Delete-and-Conquer overhead for probabilistic feedback

(b) Feedback bits (normalized) for Delete-and-Conquer

Fig. 6. Coding overhead and transmitted feedbacks (normalized by the number of inputs) by Delete-and-Conquer with *probabilistic feedback* ($c = 0.9$, $\delta = 0.1$).

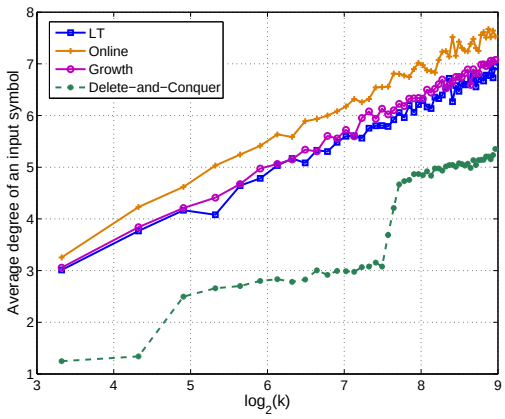


Fig. 7. Average input symbol degree for various codings ($c = 0.9$, $\delta = 0.1$).

VII. Conclusion

In this paper, we have developed Delete-and-Conquer rateless codes based on distance-type feedback messages. For very short block lengths, we have derived the precise performance gains, amount of feedback sent, and the regime in which it is economical to utilize feedback with our method. The same approach can be easily extended to obtain optimal degree distributions, minimizing the overall cost of transmissions in our method. For larger block lengths, we have experimentally validated that Delete-and-Conquer codes require less forward transmissions than standard LT codes, Growth codes, and low-overhead online codes. Moreover, we showed that Delete-and-Conquer decreases the computational complexity of encoding/decoding by excluding the recovered symbols from subsequent transmissions, in addition to intermediate performance improvement for relatively small block lengths, e.g., $k = 128$.

On the whole, we believe that Delete-and-Conquer codes can improve several aspects of rateless codes performance with a nominal utilization of the back channel (i.e., one bit per feedback for a fraction of received symbols), making them suitable for applications with constrained feedback channel. In future works, we plan to investigate the applicability of the distance-type feedback to Index Coding (IC).

References

[1] Michael Luby. LT codes. In *Annual Symposium on Foundations of Computer Science*, pages 271–280, 2002.

[2] Amin Shokrollahi. Raptor codes. *IEEE Trans. Inf. Theory*, 52(6):2551–2567, 2006.

[3] Amos Beimel, Shlomi Dolev, and Noam Singer. RT oblivious erasure correcting. *IEEE/ACM Trans. on Networking*, 15(6):1321–1332, 2007.

[4] Andrew Hagedorn, Sachin Agarwal, David Starobinski, and Ari Trachtenberg. Rateless coding with feedback. In *IEEE INFOCOM*, pages 1791–1799, 2009.

[5] Jesper Hemming Sørensen, Toshiaki Koike-Akino, and Philip Orlik. Rateless feedback codes. In *IEEE International Symposium on Information Theory*, pages 1767–1771, 2012.

[6] Abhinav Kamra, Vishal Misra, Jon Feldman, and Dan Rubenstein. Growth codes: Maximizing sensor network data persistence. In *ACM SIGCOMM*, volume 36, pages 255–266, 2006.

[7] Yuval Cassuto and Amin Shokrollahi. On-line fountain codes for semi-random loss channels. In *Information Theory Workshop (ITW), 2011 IEEE*, pages 262–266, 2011.

[8] Robert G Gallager. *Low density parity check codes*. PhD thesis, Massachusetts Institute of Technology, 1960.

[9] Claude Berrou, Alain Glavieux, and Punya Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. In *Proceeding of the IEEE International Conference on Communications*, volume 2, pages 1064–1070, 1993.

[10] John W Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege. A digital fountain approach to reliable distribution of bulk data. In *ACM SIGCOMM*, volume 28, pages 56–67, 1998.

[11] David JC MacKay. Fountain codes. In *Communications, IEE Proceedings*, volume 152, pages 1062–1068, 2005.

[12] Esa Hyytia, Tuomas Tirronen, and Jorma Virtamo. Optimal degree distribution for LT codes with small message length. In *IEEE INFOCOM*, pages 2576–2580, 2007.

[13] Ali Talari and Nazanin Rahnavard. Rateless codes with optimum intermediate performance. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6, 2009.

[14] Weiyao Xiao, Sachin Agarwal, David Starobinski, and Ari Trachtenberg. Reliable wireless broadcasting with near-zero feedback. In *IEEE INFOCOM*, pages 1–9, 2010.

[15] Rajeev Motwani. *Randomized algorithms*. Cambridge university press, 1995.

[16] Sheldon M Ross. *Introduction to probability models*. Access Online via Elsevier, 2006.