

RT Oblivious Erasure Correcting

Amos Beimel, Shlomi Dolev, Noam Singer

Abstract—An erasure correcting scheme is *rateless* if it is designed to tolerate any pattern of packet loss and reveal the transmitted information after a certain number of packets is received. On the one hand, transmission schemes that use rateless erasure correcting schemes do not usually use a feedback channel. However, they may require significant amount of additional processing by both the sender and the receiver. On the other hand, automatic repeated request protocols use a feedback channel to assist the sender, and do not usually require information processing.

In this work we present a combined approach, where a lean feedback channel is used to assist the sender to efficiently transmit the information. Our *Real-Time oblivious approach* minimizes the processing time and the memory requirements of the receiver and, therefore, fits a variety of receiving devices. In addition, the transmission is *real-time* where the expected number of original packets revealed when a packet is received is approximately the same throughout the entire transmission process.

We use our end-to-end scheme as a base for broadcast (and multicast) schemes. An overlay tree structure is used to convey the information to a large number of receivers. Moreover, the receivers may download the information from a number of senders or even migrate from one sender to another.

Index Terms—Stochastic processes, Combinatorics, Information theory, ARQ, data-link, transport-layer.

I. INTRODUCTION

IN recent years, communication networks (Internet) have been used for new applications beyond email and file transmission. The vision of an integrated telephone network (wired and/or wireless), television cables (possibly satellites supported), and the Internet (communicating data and multimedia information) in one entity is a matter of (a short period of) time. New technologies for supporting this emerging infrastructure are being investigated (e.g., [1]–[3]). One core aspect of any communication network is the way it handles message corruption. Quick and efficient message corruption handling becomes even more important in the above mentioned new applications and infrastructures.

Two main approaches are used to handle message corruption: retransmission and erasure codes. Retransmission, as used by Automatic Repeat Request (ARQ) protocols, requires little computation; however, it requires an extensive use of a

feedback channel. Forward erasure correcting schemes were suggested to cope with losses (message corruption) when multimedia information is transmitted (e.g., [4]–[11]). While forward erasure correcting schemes require no feedback, they may require an additional amount of processing by both the sender and the receiver. In this work we present a combined approach: a lean feedback channel is used to assist the sender to efficiently transmit the information.

A. Erasure Correcting and ARQ Protocols

Traditional networks use error detection codes (e.g., CRC) and Automatic Repeat Request (ARQ) protocols to identify message corruption, discard the corrupted messages, and retransmit [3]. These protocols are usually computation efficient and do not require information processing. This approach fits information that has no real-time attributes, in other words, the period of time it takes the information to reach the receiver is not too important. The receiver should receive the sent information in order, with no duplications or omissions. The ARQ approach, in which a round trip delay is associated with packet loss, does not fit transmission of multimedia information, where the transmission should satisfy time constraints.

One major limitation of ARQ protocols is the extensive use of the feedback channel in order to update the sender on the receiver's reception state. This is problematic since communication networks that support transmission of multimedia are in most cases asymmetric. The up-link from clients to the server is used for transmitting only a few control bits or may not exist altogether. For example, the up-link in cable networks is a shared relatively low rate link of about 1.5Mbps. The up-link to satellites does not usually exist or is based on a dialup modem of 56Kbps. The asymmetric property of DSL implies a much lower upstream rate than the downstream rate.

Different techniques for avoiding retransmissions were proposed in order to reduce the need for upstream. Most solutions use erasure codes, in which more data is sent on the downstream and used for correcting a reasonable amount of lost information. In general, if a sender wishes to transmit k data packets, it will instead transmit $n > k$ packets; the receiver can reconstruct the original k packets from any received subset of size at least k of the transmitted packets. Roughly speaking, erasure correcting schemes partition the data-stream into a fixed small number of k packets forming a single *window*. Each window is encoded by $n > k$ packets. The sender transmits these packets to the receiver. The ratio of n and k is chosen according to the channel success transmission rate R .

We note that partitioning the data into windows imposes some drawbacks: data is buffered both at the sender and the receiver since encoding and decoding are computed over a window of k packets. This type of buffering may delay the

©2007 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

This work is partially supported by the STRIMM consortium, IBM faculty award, NSF grant, Rita Altura trust chair in computer science, and the Lynn and William Frankel Center for Computer Science. An extended abstract of the paper appeared in PODC (Aug.) 2004, and IEEE Information Theory Workshop, San Antonio, Texas, 2004.

The authors are with the department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel.

decoding of packets with up to about n packets at the receiver's side. Furthermore, the success transmission rate R can change over time, and even if it is fixed in average, a burst of errors of about $n - k + 1$ packets may, in some cases, result in the loss of the entire window. The above phenomenon is addressed either by retransmitting the same window forever (e.g., [12], in which new comers may "drink" enough packets from a "fountain" in their own pace in order to reveal the original windows) or by coupling the erasure correcting with a limited acknowledgment (e.g., [8]). It turned out that the runtime and the space overhead of an erasure correcting scheme usually grow, while the amount of information the receiver needs for reconstructing the original data packets shrinks.

One of the classic erasure (and error correcting) codes is the *Reed Solomon* code [13]. This code is based, roughly speaking, on the fact that every k values define a polynomial function of degree $k - 1$. Thus, any k values received from a set of $n > k$ values can be used to reveal the original k values. Reed Solomon code supports the reconstruction of the original window by any k received packets but requires encoding time complexity of $O(nk)$ and decoding time complexity of $O(k^2)$. The super-linear runtime for encoding and decoding implies a choice of small values for k and n in order to keep a reasonable computing time.

Another family of erasure codes is exclusive-or based schemes. Parity bit and Hamming codes are probably one of the first schemes used for error detection/correction; both are based on using the exclusive-or function.¹ Several new techniques use the property of the exclusive-or function with additional (sometimes sophisticated) techniques for achieving erasure correcting codes [4]–[7], [9]–[11], [14]. The receiver is able to reconstruct the original k packets of the data-stream window, usually, after receiving *more* than k packets of the window. The advantages of these codes is that the computation time is usually kept approximately linear. Thus, we can characterize the above erasure codes by the trade of linear time complexity for encoding and decoding, with the expected $(1 + \epsilon)k$ number of packets required by the decoder for decoding the window. We note that, for each of these codes, the value of ϵ may be computed according to the code construction. In several of these codes, a decoding skeleton graph is reconstructed and transmitted in every encoding decoding session.

Recent schemes presented in [10], [11], [14], [15] achieve rateless codes in which the original k packets may be encoded by any number of encoding packets, thus, eliminating the setup phase in which the sender and the receiver agree, prior to the communication session, upon certain settings. In [10], the reconstruction of the original k packets can be decoded with probability $1 - \delta$ from any $k + O(\sqrt{k} \ln^2(k/\delta))$ in time complexity of $O(k \ln(k/\delta))$. In [11], [14], [15], the decoding of the message requires that approximately $(1 + \epsilon)k$ encoding packets reach the decoder, and achieves time complexity that is linear in k (for $\epsilon = 0.06$).

¹One may generalize the use of a single parity bit in a vector of n bits to $2\sqrt{n}$ bits in the case the n bits are (logically represented) by a two dimensional matrix, or even using higher dimensional array representation, up to hypercube of dimension $\log n$.

B. Our Contributions

We present a very simple protocol called RT-Oblivious erasure correcting, which is exclusive-or based. Similarly to [10], in our protocol, in order to transmit a message composed of packets (also called symbols), the sender repeatedly chooses a subset of the packets, exclusive-ors them, and sends this exclusive-or value to the receiver. If the receiver has obtained all the packets in the subset except for exactly one, then the receiver computes this new bit from the bits it already knows and the exclusive-or value. If the receiver has not obtained all the packets in the encoding except for one, then it discards this encoding. This simple discarding rule is the difference between our protocol and the previous schemes. The performance of our real-time oblivious erasure correcting scheme is controlled by the way the sender chooses the size of the subset of packages for each encoding throughout the transmission session. This size depends on the number of message packets the receiver has already obtained. The receiver uses the feedback channel to update the sender each time it has obtained sufficiently many new packets.

Our scheme has some attractive features. First, it is rateless as it can handle any channel rate. Second, using the exclusive-or operation results in a protocol with efficient computation by the sender and the receiver. Third, the discarding rule results in a very low memory overhead in both the encoder and the decoder. Namely, the decoder needs to maintain a single bit for each of the k packets that it should decode (for each packet it only needs to remember if it is already decoded). This should be compared to the previous erasure correcting codes, where the memory overhead was bigger than the size of the original message. Forth, the policy the user employs to update the sender results in low feedback (for a message with k packets, the total size of the feedback messages is $O(\sqrt{k})$). This is an important advantage in comparison to ARQ protocols. We show that, although we use a very simple scheme with important advantages, the price we pay in the transmission rate is not too high.

In our proposed protocol the expected number of symbols the decoder has to receive in order to successfully decode the data is less than twice the original data size k (experiments indicate $1.87k$). Furthermore, the probability that the expected number of encoding needed to decode the entire message is more than $2(1 + \epsilon)k$ is exponentially small for every $\epsilon > 0$. The number of required encodings in our protocol is somewhat larger than in the previous schemes; Tornado codes [5], [6], [9], on-line codes [11], [15], and Raptor codes [14] require receiving $(1 + \epsilon)k$ symbols (for, e.g., $\epsilon = 0.06$), while LT Codes [10] require receiving $k + O(\sqrt{k} \ln^2 \frac{k}{\delta})$ symbols with probability $1 - \delta$. We also note that the feedback channel reliability may influence the performance of our protocol. However, beyond eliminating the memory overhead and achieving real-time decoding, our protocol can decode the window, receiving a much smaller number of encoding packets (using several heuristics that are detailed in the sequel) in the common case in which the loss probability is stable.

Another advantage of our scheme is related to the change in the decoding rate. We use the term *real-time decoding* for

a decoding process in which the expected number of original packets revealed when a packet is received is approximately the same throughout the entire transmission process. Current erasure correcting schemes do not have the real-time property. In both the original Reed Solomon scheme and the exclusive-or based schemes the decoding computing process can only be carried out when enough encodings are aggregated at the receiver. Thus, the decoding algorithm activity is concentrated at some point of time possibly stalling the other activities of the receiver. In particular, Reed Solomon codes require that exactly k packets reach the decoder before the decoding process starts. In the case of exclusive-or based schemes (e.g., [5], [6], [9]–[11], [14], [15]) some of the packets are decoded during the transfer process as each encoding of degree one (i.e., a packet that is not exclusive-ored with any other packet) may trigger the decoding of other packets. Still, our experiments show that the majority of the packets are decoded simultaneously when no more encoded packets are needed. Thus, again, the majority of the decoding process is held when the last missing packet arrives.

The real-time property has benefits beyond processing considerations; for example, when a raster image is transferred, the image is gradually revealed. For broadcast/multicast the real-time property allows devices with low resolution to stop receiving packets for a window before devices with high resolution display stop decoding.

To summarize, the previous exclusive-or based codes, e.g., in [5], [6], [9]–[11], [14], [15], have nearly optimal rate, despite restricting themselves to the simple exclusive-or operation. This fact justifies the restriction to the exclusive-or operation. Furthermore, the exclusive-or operation has the additional attractive advantage that it is extremely efficient and is simpler than operations over larger fields. Our restriction that packets that cannot be immediately decoded are discarded does, however, lead to a loss in achievable rates. However, the resulting scheme is very simple and has attractive properties.

We use our end-to-end scheme as a base for broadcast (and multicast) schemes. An overlay tree structure is used to convey the information to a large number of receivers. Moreover, the receivers may download the information from a number of senders or even migrate from one sender to another while receiving encoded packets of a certain window. Independently, (though slightly later), Lun et al. [16] presented schemes to transfer data to a general set of receivers; specifically they present a scheme based on acyclic graphs (and not only trees). Their schemes do not use feedback messages. Our scheme is based on exclusive-or as opposed to the computational more expensive method used in [16]. That is, the decoding complexity in our scheme is linear, whereas theirs is quadratic. The second advantage of our scheme is the low memory overhead. Online encoding for nodes in the middle, as well as for leaves, has been introduced in the setting of network coding [17] and has been studied extensively.

C. Paper organization

The setting of the communication network that we consider appears in Section II. The basic real-time erasure correcting

scheme is described in Section III, and its analysis is given in Section IV. Extensions to the case of broadcast/multicast tree and other applications are discussed in Section V. Several heuristics methods for achieving more efficient performances are specified in Section VI.

II. NETWORK SETTINGS

We assume there is a *sender* (also called *encoder*) and a *receiver* (also called *decoder*) that are connected by a full duplex asymmetric *communication link*. The communication link is composed of a *down-link* that transfers information from the sender to the receiver and an *up-link* that transfers information from the receiver to the sender. The communication is asymmetric in the sense that the amount of information transmitted through the up-link is bounded by a (small) fraction of the amount of information transmitted through the down-link. The information is sent and received in the form of a bounded stream of bits called *packet*. In the sequel, we use the term *symbol* instead of packet, as each possible packet content can be viewed as a symbol from a given alphabet.

In our settings, a symbol sent on the down-link may be lost without any assumption on the loss pattern, i.e., there is no assumption on the channel rate and an adversary can arbitrarily choose the symbols that are lost. The only restriction is that the adversary cannot check the content of the packets. We will show that, under these weak assumptions, the expected number of symbols needed to decode the entire message is relatively small.²

An input *window* x_1, \dots, x_k is a sequence of size k (input) packets. Our scheme is exclusive-or based, where bit-wise exclusive-or is performed on packets of a window. The packets sent by the sender are called *encodings*. Each encoding packet is an exclusive-or of a subset of the window packets. The *degree* of an encoding, denoted by d , is the number of symbols that are used to create the encoding (via the exclusive-or operation). A symbol is *revealed* (by the receiver) if the receiver decodes it using the decoding procedure. The process of decoding a symbol is also called *revealing* a symbol. The number of the symbols the receiver has revealed/decoded will be denoted by r for the remaining part of this paper.

In Section V, we also consider a more complicated network model in which there are several receivers that are represented by nodes of a tree rooted at a sender, where each neighboring nodes communicate using our oblivious protocol and the nodes collaborate in the process of distributing the information (more details appear in Section V).

III. REAL-TIME OBLIVIOUS ERASURE CORRECTING

In our scheme, similarly to [10], the encoder repeatedly chooses a degree d , in a specific manner detailed in the sequel, and then randomly chooses d different symbols $x_{\pi_1}, x_{\pi_2}, \dots, x_{\pi_d}$ from the input window. Subsequently, the sender bit-wise exclusive-ors the bits of the chosen symbols to obtain the encoding symbol e . Namely, $e = \bigoplus_{i=1}^d x_{\pi_i}$. We say that $x_{\pi_1}, x_{\pi_2}, \dots, x_{\pi_d}$ are the *generating symbols* of e .

²First, we assume that symbols sent by the decoder on the up-link reach the encoder reliably and without delay, later we relax this assumption.

The sender sends e together with the indices $\pi_1, \pi_2, \dots, \pi_d$ to the receiver (in the sequel we use identical seeds and a pseudo-random generator to eliminate sending the explicit list of indices, see Section VI-C). Since the exclusive-or operation is reversible, a decoder which receives e can reconstruct one generating symbol of e when a subset of exactly $d - 1$ generating symbols are already revealed.³ If all the generating symbols of e have been revealed previously, then no information is added to the decoder. Furthermore, if more than one generating symbol is missing, the decoder may either record e for possible future use (using auxiliary data memory) or discard e (allowing small memory and processing overheads). Recorded encoding symbols are usually maintained using a bipartite-graph data structure in order to facilitate future decoding [5], [6], [9]–[11].

We consider a simple oblivious scheme in which any encoding that does not have exactly one unrevealed generating symbol is discarded. Revealing succeeds when the decoder receives an encoding e whose generating symbols are $x_{\pi_1}, \dots, x_{\pi_d}$, and the decoder has already revealed all generating symbols except x_{π_j} . The decoding of x_{π_j} is computed as the exclusive-or of e with all the generating symbols of e which have already been revealed.

The performance of our real-time oblivious erasure correcting scheme is controlled by the way the sender chooses the degree d for each encoding throughout the transmission session. In the sequel, we present an algorithm for the sender to compute the current optimal degree of an encoding, given the number of symbols that had already been revealed by the receiver. The goal is to maximize the probability that the computed encoding is used by the receiver to reveal a symbol (of the window), in other words, to maximize the probability that all but exactly one generating symbol of the encoding are previously revealed. In Section IV, we analyze the maximal revealing probability and the degree, denoted $d(r)$, that achieves this probability.

In more details, we assume that at each point of the transmission session, the sender knows (using the feedback channel) the number of symbols the receiver has currently revealed, denoted r , and computes a degree $d(r)$ for the next encoding. The encoder then randomly picks $d(r)$ different symbols from the original symbols, computes the bit-wise exclusive-or of these symbols and sends the result as the next encoding. We show that $d(r) = \lfloor (k+1)/(k-r) \rfloor$ is the optimal degree for maximizing the revealing probability.

The receiver maintains a bit for every original symbol indicating whether the symbol has been already revealed. When the decoder receives a new encoding e , the receiver first scans the list of indices of the generating symbols of e . The decoder checks that exactly one of these indices is of an unrevealed symbol, and if so, decodes the value of this symbol by computing the exclusive-or of the encoding with all the rest of the generating symbols which had previously been revealed, i.e., $x_{\pi_j} = e \oplus (\oplus_{i \neq j} x_{\pi_i})$.

The receiver updates the sender with its current decoding

³An alternative approach is solving linear equations over $\text{GF}(2)$, which may involve super-linear complexity.

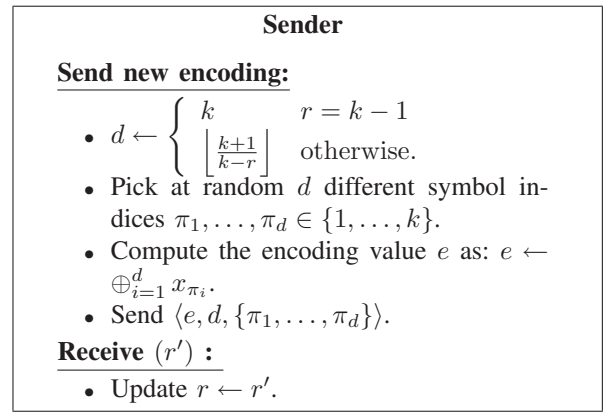


Fig. 1. The RT protocol of the sender.

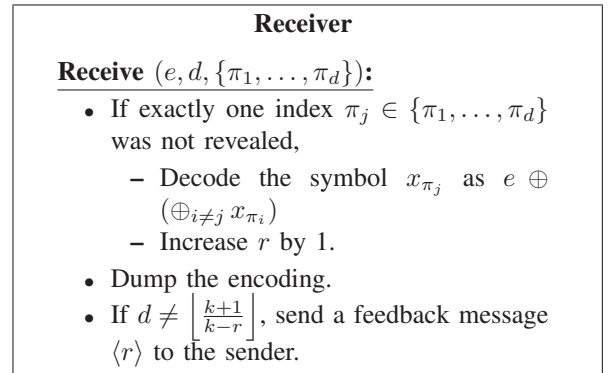


Fig. 2. The RT protocol of the receiver.

status by sending the number of revealed symbols r , either upon each revealing of a symbol, or only whenever the degree the sender should use changes, that is, when $\lfloor (k+1)/(k-r) \rfloor$ changes.

Fig. 1 describes the sender protocol for sending encoding symbols and updating the currently known number of revealed symbols r at the receiver's side upon reception of a new feedback message. Fig. 2 describes the protocol of the receiver for receiving an encoding, decoding a symbol (if possible), and updating the sender with the current number of symbols revealed so far by the receiver.

IV. ANALYSIS OF THE PROTOCOL

In this section we will analyze the overall efficiency of our protocol, as summarized in the next theorem.

Theorem 4.1: The expected number of encodings required for decoding the entire message is less than $2k$, and for every $0 < \epsilon < 1$ the probability that $(1 + \epsilon)2k$ encodings are not sufficient for decoding the entire message is less than $e^{-O(\epsilon^2 k)}$. The expected number of feedback messages is $O(\sqrt{k})$, and the total expected decoding complexity is $O(k \log k)$.

A. The Optimal Degree

Towards the proof of Theorem 4.1, we will argue that the probability that an encoding reveals an original symbol is at least $1/e$, provided that the encoder chooses the optimal

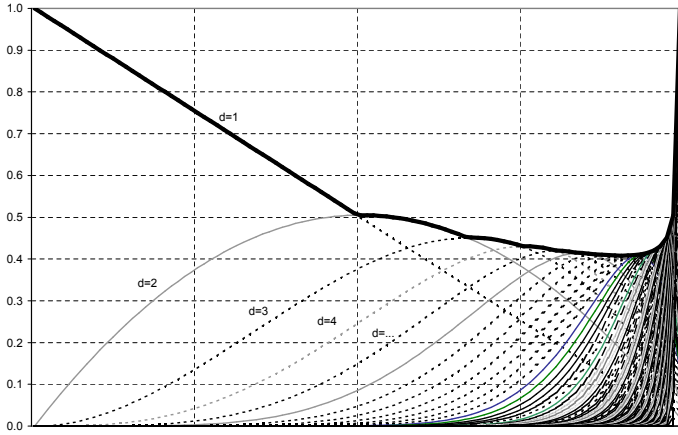


Fig. 3. The revealing probability as a function of the number of revealed symbols r with $k = 100$ transmitted data packets for the various values of the degrees. The x -axis denotes the number of revealed symbols r and the y -axis denotes the probability $P(d, r)$.

degree. To establish the bound on the decoding probability, we first prove that the optimal degree is $\lfloor (k+1)/(k-r) \rfloor$.

Definition 4.2: The *revealing probability* of an encoding is the probability that an input symbol is decoded as a result of the reception of an encoding, that is, iff exactly one of the generating symbols of the encoding had not been previously revealed. We denote the revealing probability of an encoding of degree d , when r (of a given k) symbols had been already revealed, by $P(d, r)$.

To build our intuition, we first examine some specific cases. The probability that the receiver will decode a new symbol when the encoding symbol is generated from a single input symbol and the receiver has not revealed any symbols is exactly one, that is, $P(d=1, r=0) = 1$. In the other extreme, the probability that the receiver will decode a new symbol when the encoding symbol is generated from k input symbols and the receiver has not revealed exactly one symbol is one (that is, $P(d=k, r=k-1) = 1$). This is due to the fact that in this case the only symbol that had not been revealed can be revealed using all the remaining symbols and the encoding symbol. One more simple case is $P(d, r) = 0$ for $d \geq r+2$, as there must be at least two missing symbols. Finally, $P(d=1, r) = (k-r)/k$, as the encoding symbol is a single input symbol, which can be used by the decoder iff the symbol has not been previously revealed. We next analyze the revealing probability for all values of d and r . The following lemma was proven in [10].

Lemma 4.3: For $0 \leq d \leq r+1 \leq k$, the revealing probability satisfies

$$P(d, r) \triangleq \Pr[\text{revealing}] = \frac{d(k-r)}{k} \prod_{j=1}^{d-1} \frac{r+1-j}{k-j}.$$

Our goal is to find the optimal degree that maximizes the revealing probability, assuming knowledge of the number of revealed symbols r by the receiver. There are exactly k possible values for d . Therefore there are k different probabilities, and we seek the maximal probability for each value of r .

Fig. 3 depicts the probability function $P(d, r)$ as analyzed in Lemma 4.3 for every possible value of d where $k = 100$.

The probability is depicted as a function of r , the number of symbols that had already been revealed by the receiver. For each r , there exists an encoding degree d that maximizes the revealing probability.

To make our protocol efficient, we would like to maximize the revealing probability. Thus, given k we should find the best d that maximizes $P(d, r)$.

Definition 4.4: The *optimal probability* for a given r and k is defined as $P(r) \triangleq \max_{1 \leq d \leq k} \{P(d, r)\}$. The *optimal degree* $d(r)$ is a degree which maximizes the revealing probability for r , i.e., $P(d(r), r) \geq P(d, r)$ for every $1 \leq d \leq k$.

The next lemma gives a simple formula for $d(r)$.

Lemma 4.5: For every $r \in \{0, \dots, k-1\}$, the optimal degree $d(r)$ is

$$d(r) = \begin{cases} \lfloor \frac{k+1}{k-r} \rfloor & r < k-1, \\ k & r = k-1. \end{cases}$$

Proof: We will show that for a given r the function $P(d, r)$ increases when $d < \lfloor (k+1)/(k-r) \rfloor$ and decreases when $d > \lfloor (k+1)/(k-r) \rfloor$. Thus, $d = \lfloor (k+1)/(k-r) \rfloor$ is the optimal value for d . We first examine when $P(d, r) - P(d-1, r) > 0$.

$$\begin{aligned} & P(d, r) - P(d-1, r) \\ &= \frac{k-r}{k} \prod_{j=1}^{d-1} \frac{r+1-j}{k-j} \left(d - (d-1) \frac{k-d+1}{r-d+2} \right) \\ &= \frac{k-r}{k} \prod_{j=1}^{d-1} \frac{r+1-j}{k-j} \left(\frac{d(r-k) + (k+1)}{r-d+2} \right). \end{aligned}$$

Since $d \leq r+1 \leq k$, the denominator $r-d+2$ is always positive. Therefore, the value of $P(d, r) - P(d-1, r)$ is positive iff the numerator $d(r-k) + (k+1)$ is positive. Thus, $P(d, r) > P(d-1, r)$ iff $d < \frac{k+1}{k-r}$. This means that $P(d, r)$, as a function of d for a fixed r , increases when $d < \frac{k+1}{k-r}$, decreases when $d > \frac{k+1}{k-r}$, and does not change if $d = \frac{k+1}{k-r}$. Therefore, an optimal degree $d(r)$ is $\lfloor \frac{k+1}{k-r} \rfloor$.⁴ When $r = k-1$, the function increases in the range $\{1, \dots, k\}$, and its maximum is attained when $d = k$. ■

Fig. 4 depicts the optimal encoding degree $d(r)$ for all possible values of r ranging from 0 to $k-1$, where k is chosen to be 100. By Lemma 4.5, the optimal degree for $r < k/2$ is one. Note that Fig. 3 also supports the above observation, since $P(d=1, r)$ is maximized at that range. Furthermore, when $r = k-1$ the optimal degree is k and the corresponding encoding is the exclusive-or of all symbols; this encoding always enables the reconstruction of the last symbol.

B. The Optimal Revealing Probability

To bound the revealing probability from below, we define a continuous function that is analogous of the discrete function $P(d, r)$.

⁴If $d = \frac{k+1}{k-r}$ is an integer then $P\left(\frac{k+1}{k-r}, r\right) = P\left(\frac{k+1}{k-r} - 1, r\right)$ and $d = \frac{k+1}{k-r} - 1 = \frac{r+1}{k-r}$ is an optimal degree as well.

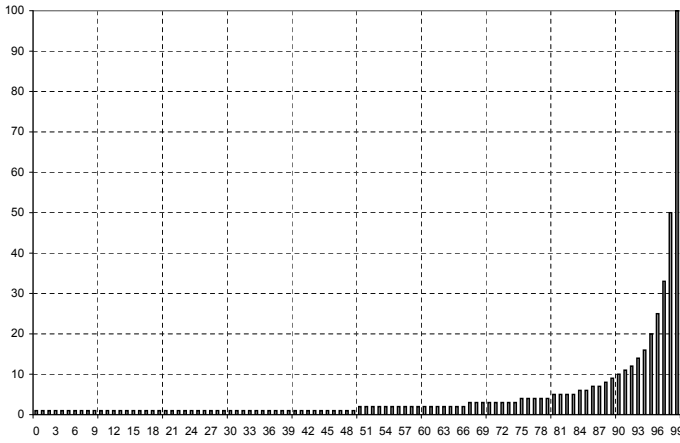


Fig. 4. Optimal encoding degree for $0 \leq r \leq k-1$, for $k=100$. The x-axis denotes the number of revealed symbols r and the y-axis denotes the optimal degree $d(r)$.

Definition 4.6: We define $\hat{P}_d(r) \triangleq \frac{d(k-r)}{k} \prod_{j=1}^{d-1} \frac{r+1-j}{k-j}$ for every real number $r \in [d-1, k-1]$.

Lemma 4.7: Fix a degree $d \in \{1, \dots, k\}$ and let $d-1 \leq a < b \leq k-1$. The function $\hat{P}_d(r)$, as a function of r in the subrange $[a, b]$, has a minimum at $r = a$ or at $r = b$ (or both). That is, $\hat{P}_d(r) \geq \min\{\hat{P}_d(a), \hat{P}_d(b)\}$ for every $r \in [a, b]$.

Proof: We first consider $\hat{P}_1(r) = \frac{k-r}{k}$. For any range $[a, b]$, the minimum of $\hat{P}_1(r)$ is obtained when $r = b$ as $\hat{P}_1(r)$ is monotonically decreasing.

We next consider $\hat{P}_d(r)$ for $d > 1$ and consider its derivative

$$\begin{aligned} \hat{P}_d(r)' &= \left(d \frac{k-r}{k} \prod_{j=1}^{d-1} \frac{r+1-j}{k-j} \right)' \\ &= \frac{d}{k \prod_{j=1}^{d-1} (k-j)} \left((k-r) \prod_{j=0}^{d-2} (r-j) \right)' \\ &= \hat{P}_d(r) \cdot \left(\sum_{i=0}^{d-2} \frac{1}{r-i} - \frac{1}{k-r} \right). \end{aligned}$$

We define $g(r)$ as the right product of $\hat{P}_d(r)'$, that is, $g(r) \triangleq \left(\sum_{i=0}^{d-2} \frac{1}{r-i} - \frac{1}{k-r} \right)$. The value of $\hat{P}_d(r)$ is always positive for $r \in [d-1, k-1]$; thus in order to analyze the local extremum of $\hat{P}_d(r)$ we examine the roots of $\hat{P}_d(r)'$; these roots are the roots of $g(r)$. We first compute $g(r)' = \sum_{i=0}^{d-2} \frac{-1}{(r-i)^2} - \frac{1}{(k-r)^2} \leq 0$.

Since $r \in [d-1, k-1]$ and, in particular, $r \neq k$, we conclude that $g(r)' < 0$. Thus, $g(r)$ is monotonically decreasing, and $\hat{P}_d(r)$ has at most one root in the range. Therefore, $\hat{P}_d(r)$ has no more than one local extremum in $(d-1, k-1)$.

We show that if such extremum point exists, it must be a maximum point. Specifically, we show that the function $\hat{P}_d(r)$ initially increases by showing that for the minimal r value $\hat{P}_d(r) > 0$, which follows from showing that $g(d-1) = \sum_{i=0}^{d-2} \frac{1}{(d-1)-i} - \frac{1}{k-(d-1)} \geq 1 - \frac{1}{k-(d-1)} \geq 0$.

Thus, for $r = d-1$, which is the minimal value of r , the value of $g(r)$ is positive and $\hat{P}_d(r)$ is also positive. Furthermore, $\hat{P}_d(r)$ reaches a maximum when $g(r) = 0$, which can occur no more than once since $g(r)'$ is always negative.

This proves that if $\hat{P}_d(r)$ has a single maximum in the range $[d-1, k-1]$, then for any subrange $[a, b] \subseteq [d-1, k-1]$ a minimum can be reached for $r = a$ or for $r = b$. However, if $\hat{P}_d(r)$ has no local maximum in the range $[d-1, k-1]$ then it is monotonically increasing, thus for any subrange $[a, b]$ a minimum is reached for $r = a$. ■

As seen in Fig. 3, for $k=100$, the optimal revealing probability reaches a minimum of about 0.4. The following lemma shows that for any message with k symbols, the minimal revealing probability is more than $1/e$.

Lemma 4.8: For any $0 \leq r \leq k-1$ the optimal revealing probability when using the optimal degree $d(r) = \lfloor (k-1)/(k-r) \rfloor$ satisfies $P(r) > 1/e$.

Proof: Instead of analyzing the discrete function $P(r)$, we will analyze its extension as a continuous function

$$\hat{P}(r) \triangleq \max_{1 \leq d \leq k-1} \hat{P}_d(r).$$

For any integer r , by definition $\hat{P}(r) = P(r)$, therefore it is enough to prove that $\hat{P}(r) \geq 1/e$ for every $r \in [0, k-1]$. In order to analyze this function, we will find for each d the range $[a_d, b_d]$ in which $\hat{P}(r) = \hat{P}_d(r)$ and use Lemma 4.7 which proves that the minimum of $\hat{P}_d(r)$ in this range is reached at one of its end points a_d, b_d . We will show that the values of $\hat{P}_d(r)$ at the points a_d, b_d are larger than $1/e$. See Fig. 3 for illustration.

By the same arguments as in the proof of Lemma 4.5, it holds that $\hat{P}(r) = \hat{P}_{d(r)}(r)$, where $d(r) = \lfloor (k+1)/(k-r) \rfloor$. Thus, the optimal degree $d(r)$ is a non-decreasing function. Furthermore, when $d(r)$ increases, it increases by exactly 1. This implies that $b_d = a_{d+1}$ and $(k+1)/(k-a_{d+1}) = d+1$, which implies that $a_{d+1} = \frac{dk-1}{d+1}$. Furthermore, $\hat{P}_d\left(\frac{dk-1}{d+1}\right) = \hat{P}_{d+1}\left(\frac{dk-1}{d+1}\right) = \hat{P}\left(\frac{dk-1}{d+1}\right)$. To conclude, by Lemma 4.7, for every $r \in [0, k-1]$

$$\begin{aligned} \hat{P}(r) &= \hat{P}_{d(r)}(r) \\ &\geq \min\left\{\hat{P}_{d(r)}(a_d(r)), \hat{P}_{d(r)}(a_{d(r)+1})\right\} \\ &= \min\left\{\hat{P}_{d(r)-1}(a_d(r)), \hat{P}_{d(r)}(a_{d(r)+1})\right\}. \end{aligned}$$

To complete the proof we need to show that $\hat{P}_d(a_{d+1}) = \hat{P}_d\left(\frac{dk-1}{d+1}\right) > 1/e$ for every $d \in \{1, \dots, k\}$.

$$\begin{aligned} \hat{P}_d\left(\frac{dk-1}{d+1}\right) &= d \frac{k - \frac{dk-1}{d+1}}{k} \prod_{j=1}^{d-1} \frac{\frac{dk-1}{d+1} + 1 - j}{k-j} \\ &\geq \frac{d}{d+1} \left(\frac{\frac{dk-1}{d+1} + 2 - d}{k+1-d} \right)^{d-1} \\ &\geq \frac{d}{d+1} \left(\frac{d(k-d+1)}{(k+1-d)(d+1)} \right)^{d-1} \\ &= \left(\frac{d}{d+1} \right)^d = \frac{1}{\left(1 + \frac{1}{d}\right)^d} > \frac{1}{e}. \end{aligned}$$

The last inequality is true since $\forall x > 0, (1 + \frac{1}{x})^x < e$. ■

C. The Expected Number of Required Encodings

We are now ready to bound the expected number of symbols needed to decode the entire message.

Observation 4.9: The expected number of encoding symbols required for revealing a single symbol when using the optimal degree $d(r)$ is $\text{EXPECTED}(r) \triangleq \frac{1}{P(r)} \leq e$.

Observation 4.9 implies that the expected total number of encodings required to decode the entire message of k symbols is at most ek . The next lemma gives a better upper bound on this expected value.

Lemma 4.10: The expected total number of encodings required to decode the entire message of k symbols is at most $2 \cdot k + O(1)$.

Proof: First, $P(r) \geq P(1, r) = (k - r)/k$ for every $0 \leq r \leq k - 1$. Second, by Lemma 4.8, for any value of r the revealing probability is always larger than $1/e$.

We will need some properties of H_n – the harmonic sum – defined as $H_n \triangleq \sum_{i=1}^n 1/i$. By [18, Equation 6.66], there exists a constant $0 < \gamma < 1$ and for every n there exists a constant $0 < \epsilon_n < 1$ such that the harmonic sum is approximated by $H_n = \ln n + \gamma + \frac{1}{2n} - \frac{1}{12n^2} + \frac{\epsilon_n}{120n^4}$. Thus, $\ln n + \gamma < H_n < \ln n + \gamma + \frac{1}{2n}$. Furthermore, one can check that $-\ln(1 - x) \leq 2x$ for $0 < x < \frac{1}{2}$.

We next bound the expected number of symbols needed to decode the k original symbols.

$$\begin{aligned}
\text{EXPECTED} &= \sum_{r=0}^{k-1} \frac{1}{P(r)} \\
&\leq \sum_{r=0}^{\lfloor k - \frac{k}{e} \rfloor} \frac{k}{k-r} + \sum_{r=\lfloor k - \frac{k}{e} \rfloor + 1}^{r=k} e \\
&\leq k \sum_{j=\lfloor \frac{k}{e} \rfloor}^k \frac{1}{j} + e \left\lceil \frac{k}{e} \right\rceil \\
&< k \left(H_k - H_{\lfloor \frac{k}{e} \rfloor - 1} \right) + k + 3 \\
&\leq k \left(\ln k - \ln \left(\frac{k}{e} - 2 \right) + \frac{1}{2k} \right) + k + 3 \\
&< k \left(\ln e - \ln \left(1 - \frac{2e}{k} \right) \right) + k + 4 \\
&\leq 2k \frac{2e}{k} + 2k + 4 = 2k + O(1). \quad \blacksquare
\end{aligned}$$

Using simulations, we have tested the expected number of required symbols for large values of k and found that in our simulations the actual factor converges to less than 1.87. Although a factor of 1.87 is rather high, for relatively reliable channels, the heuristic described in Section VI-A improves this factor significantly.

D. The Deviation of the Number of Required Encodings

We show that the probability that the number of encodings needed to decode the entire message is more than $2k(1 + \epsilon)$ is exponentially small in k .

Lemma 4.11: Let M be a random variable denoting the number of symbols required for decoding the entire message. For $0 < \epsilon < 1$,

$$\Pr [M \geq 2k(1 + \epsilon)] \leq 4 \cdot e^{-\epsilon^2 \cdot (k-1/2)/e^4}.$$

Furthermore, for $\epsilon \geq 1$

$$\Pr [M \geq 4k] \leq 2 \cdot e^{-(k-1/2)/18.493} + e^{-k/10.874}.$$

To prove the lemma we analyze the probability that $2k(1 + \epsilon)$ encodings are not sufficient for decoding the k message symbols. In order to bound this probability, we separate the decoding process into two phases.

In the first phase we consider the decoding of the first $k(1 - 1/e)$ symbols, and in the second phase we consider the decoding of the last k/e symbols. In the next two claims, we prove that in each phase the probability that $k(1 + \epsilon)$ encodings are not sufficient for decoding the symbols is small. Thus, the probability that $2k(1 + \epsilon)$ symbols are not sufficient for decoding the entire message is small. We begin by analyzing the first phase.

Claim 4.12: Let M_1 be a random variable denoting the number of symbols required for decoding the first $k(1 - 1/e)$ symbols. For $0 < \epsilon < 1$,

$$\Pr [M_1 \geq k(1 + \epsilon)] \leq 2 \cdot e^{-\epsilon^2(k-1/2)/e^4}.$$

Proof: In our protocol we decode the first $k/2$ symbols using degree $d = 1$, and then increase the degree. In our analysis we assume that the first $k(1 - 1/e)$ symbols are decoded using degree $d = 1$. Since the protocol uses the optimal degree, decoding with $d = 1$ when the degree should be increased can only decrease the probability of decoding each symbol.

The process of decoding encodings of degree $d = 1$ is equivalent to the *occupancy problem*, whose analysis can be found in [19, Theorem 4.18]. Below we describe the occupancy problem.

In the occupancy problem process, m balls are thrown into k bins. Each ball is thrown randomly with uniform distribution to one of the bins, independently of the other balls. Let $Z^{(m)}$ be a random variable denoting the number of empty bins after m balls have been thrown. E.g., $Z^{(0)} = k$, $Z^{(1)} = k - 1$, and if $Z^{(m)} = 0$, then all bins are full. By [19, Theorem 4.18], the average number $\mu^{(m)}$ of empty bins $Z^{(m)}$ satisfies:

$$\mu^{(m)} = E(Z^{(m)}) = k \left(1 - \frac{1}{k} \right)^m \leq k \cdot e^{-m/k}. \quad (1)$$

Furthermore, for every $\lambda > 0$, the probability that the expected number of empty bins differs from $\mu^{(m)}$ can be bounded by:

$$\begin{aligned}
\Pr [Z^{(m)} - \mu^{(m)} \geq \lambda] &\leq \Pr \left[\left| Z^{(m)} - \mu^{(m)} \right| \geq \lambda \right] \\
&\leq 2 \cdot e^{-\frac{\lambda^2(k-1/2)}{\bar{k}^2}}.
\end{aligned}$$

For our analysis, we set m to be $k(1 + \epsilon)$, and therefore we approximate $\mu^{(m)}$, the expected number of empty bins $Z^{(m)}$, by:

$$\mu^{(m)} \leq k \cdot e^{-(1+\epsilon)}. \quad (2)$$

We next bound the failure probability in phase 1, that is, the probability that after m encodings, the number of undecoded symbols is greater than k/e , assuming that in phase 1, the encoder is using degree $d = 1$ (this only increases the failure probability). We view sending an encoding containing a random input symbol x_i as randomly throwing a ball into the i th bin. The input symbol x_i is revealed iff the i th bin is not empty. That is, the probability that m encodings reveal at least $k(1 - 1/e)$ input symbols is equal to the probability that, after throwing m balls, at most k/e bins are empty. We next analyze this probability. (Some explanations for the inequalities are given below.)

$$\Pr \left[Z^{(m)} \geq \frac{k}{e} \right] = \Pr \left[Z^{(m)} - \mu^{(m)} \geq \frac{k}{e} - \mu^{(m)} \right] \\ \leq \Pr \left[Z^{(m)} - \mu^{(m)} \geq \frac{k}{e} - \frac{k}{e^{\epsilon+1}} \right] \quad (3)$$

$$\leq 2 \cdot e^{-\left(\frac{\epsilon-1}{e+1}\right)^2 (k-1/2)} \quad (4)$$

$$\leq 2 \cdot e^{-(\epsilon-1)^2 (k-1/2)/e^4} \quad (5)$$

$$\leq 2 \cdot e^{-\epsilon^2 (k-1/2)/e^4}. \quad (6)$$

Inequality (3) follows from Inequality (2). Inequality (5) follows from the assumption in the lemma that $\epsilon < 1$. Finally, Inequality (6) follows from the fact that for any ϵ it holds that $e^\epsilon - 1 \geq \epsilon$. ■

We next analyze the probability of failure at the second phase.

Claim 4.13: Assume that exactly $k(1 - 1/e)$ symbols are already decoded, and let M_2 be the number of symbols required for decoding the last k/e symbols. For $0 < \epsilon < 1$,

$$\Pr [M_2 \geq k(1 + \epsilon)] \leq e^{-\epsilon^2 k \frac{1}{4e}}.$$

Proof: The second phase starts when there are exactly $k(1 - 1/e)$ decoded symbols. We analyze the decoding process using encodings of the optimal degree.

For the analysis we will use the Chernoff Inequality. For m independent random variables $\{Y_1, \dots, Y_m\}$ with values 0 or 1, such that probability $\Pr [Y_i = 1] = p$, and for $Y^{(m)} = \sum_{i=1}^m Y_i$, the Chernoff inequality states that for every $0 < \lambda < 1$ the probability that $Y^{(m)} \leq (1 - \lambda)pm$ is bounded by:

$$\Pr [Y^{(m)} \leq (1 - \lambda)pm] \leq e^{-\lambda^2 pm/2}.$$

From Lemma 4.8, the probability that an encoding reveals a new symbol is at least $1/e$. We now define the random variable X_i to be 1 if the i th encoding reveals a symbol and 0 otherwise, where $1 \leq i \leq m$. (The indices of X_i are the encodings indices of the second phase only). The probability that $X_i = 1$ is greater than $1/e$, however these random variables are not independent. That is, the exact success probability depends on the degree of the encoding which depends on the number of previously decoded encodings. To overcome this technical difficulty, we define the random variables Y_1, \dots, Y_m as follow: For every $1 \leq i \leq m$, given that $X_1 = \xi_1, X_2 = \xi_2, \dots, X_{i-1} = \xi_{i-1}$, let $r_i = (1 - 1/e)k + \sum_{j=1}^{i-1} \xi_j$, that is r_i denotes the number of revealed symbols prior to the arrival of the i th encoding. By Lemma 4.5, the exact probability that the i th encoding decodes a symbol is $P(r_i)$. For completeness, define $P(k) = 1$.

Now define Y_i : If $X_i = 0$ then $Y_i = 0$ with probability 1. If $X_i = 1$ then $Y_i = 1$ with probability $1/(e \cdot P(r_i))$, and $Y_i = 0$ with probability $(1 - 1/(e \cdot P(r_i)))$. Thus,

$$\Pr [Y_i = 1 | X_1 = \xi_1, \dots, X_{i-1} = \xi_{i-1}] \\ = \frac{\Pr [X_i = 1 | X_1 = \xi_1, \dots, X_{i-1} = \xi_{i-1}]}{e \cdot P(r_i)} = 1/e$$

for any history of decoded symbols. With this construction, the random variable Y_i is independent of X_1, \dots, X_{i-1} , and therefore independent of Y_1, \dots, Y_{i-1} . Thus, Y_1, \dots, Y_m are independent random variables where $\Pr [Y_i = 1] = p = 1/e$.

We are interested in finding the probability of failing to decode the final k/e symbols using $m = k(1 + \epsilon)$ encodings which is the probability that $X^{(m)} < k/e$. The probability that $Y_i = 1$ is less than the actual decoding process. Thus, using Chernoff bound and the assumption that $0 < \epsilon \leq 1$, the probability of failure at phase 2 is at most

$$\Pr \left[Y^{(m)} \leq \frac{k}{e} \right] = \Pr \left[Y^{(m)} \leq \left(1 - \frac{\epsilon}{1 + \epsilon}\right) \cdot \frac{1}{e} \cdot m \right] \\ \leq e^{-\left(\frac{\epsilon}{1 + \epsilon}\right)^2 \cdot \frac{m}{2e}} \\ = e^{-\frac{\epsilon^2}{1 + \epsilon} \cdot \frac{k}{2e}} \\ \leq e^{-\epsilon^2 k \frac{1}{4e}}. \quad (7)$$

We now prove Lemma 4.11 by combining the two phases to show the bound on the failure probability.

Proof: The probability that more than $2k(1 + \epsilon)$ encodings are required to decode the entire message is less than the probability that more than $k(1 + \epsilon)$ encodings are required in at least one of the two phases, that is, the probability that $M_1 > (1 + \epsilon)k$ or $M_2 > (1 + \epsilon)k$. Therefore, by the union bound, the failure probability can be bounded by $2 \cdot e^{-\epsilon^2 \cdot \frac{k-1/2}{e^4}} + e^{-\epsilon^2 k \frac{1}{4e}} \leq 4 \cdot e^{-\epsilon^2 \cdot \frac{k-1/2}{e^4}}$.

We next analyze the failure probability for $\epsilon > 1$. From Inequalities (4) and (7) for $\epsilon > 1$ the probability that more than $2k(1 + \epsilon)$ encodings are required to decode the message can be bounded by:

$$\Pr [M \geq 2k(1 + \epsilon)] \leq 2 \cdot e^{-(k-1/2)/18.493} + e^{-k/10.874}.$$

E. Analyzing the Feedback and Work

In this section we analyze the number of feedback messages the decoder needs to send to the encoder. Furthermore, we analyze the work of the decoder.

There are k possible degrees for encoding. The following lemma will show that using the optimal degree (of Lemma 4.5), the encoder uses only $O(\sqrt{k})$ possible degree values. Thus, the receiver may reduce the amount of feedback messages required by the protocol, and transmit only $O(\sqrt{k})$ messages, notifying the encoder that the degree should be changed. In other words, we achieve a very lean up-link communication, which is dramatically smaller than the up-link communication used by ARQ protocols.

Lemma 4.14: For $0 \leq r \leq k - 1$ there exist no more than $2\sqrt{k + 1}$ different values for the optimal degrees $d(r)$.

Proof: First, $d(r) = \lfloor (k+1)(k-r) \rfloor$ is non decreasing for $0 \leq r \leq k-1$. Thus, for $r \leq k - \sqrt{k+1}$ the value $d(r) \leq d(k - \sqrt{k+1}) = \sqrt{k+1}$. Since the function $d(r)$ is discrete and non-decreasing, for $r \leq k - \sqrt{k+1}$ there are no more than $\sqrt{k+1}$ changes in the value of the optimal degree $d(r)$.

Furthermore, for $r > k - \sqrt{k+1}$, although the value of $d(r)$ can change on every change of r , there are exactly $\sqrt{k+1}$ values of r , thus, there are no more than $\sqrt{k+1}$ possible values of $d(r)$. The total number of possible values of $d(r)$ is, therefore, no more than $2 \cdot \sqrt{k+1}$. ■

We can optimize the feedback channel usage by sending a feedback message with the new value of r only when the value of $d(r)$ has changed, i.e., when $d(r) \neq d(r-1)$. With this optimization the amount of feedback messages is reduced to $O(\sqrt{k})$.

We will now show that the total expected processing time required in the RT Oblivious protocol by the receiver is reasonably low.

Lemma 4.15: In our RT Oblivious protocol, the expected amount of processing required from the receiver is $O(k \log k)$.

Proof: First, the complexity of handling an encoding is proportional to the degree of the encoding, as it takes $O(d)$ steps to check if exactly one symbol has not been revealed and reveal it if possible. Furthermore, the expected number of encodings required to reveal a single symbol is less than e . Therefore, the expected complexity of the receiver is

$$\begin{aligned} & O\left(\sum_{r=0}^{k-1} (\text{EXPECTED}(r) d(r))\right) \\ &= O\left(\sum_{r=0}^{k-1} \left(e \left\lfloor \frac{k+1}{k-r} \right\rfloor\right)\right) \\ &= O\left(k \cdot \sum_{r=0}^{k-1} \frac{1}{k-r}\right) \\ &= O\left(k \cdot \sum_{j=1}^k \frac{1}{j}\right) = O(k \log k). \end{aligned}$$

F. Handling the Last $\sqrt{k+1}$ Symbols

As we have seen in Lemma 4.14, for $r > k - \sqrt{k+1}$ the value of $d(r)$ can change every time the receiver reveals a symbol (in fact, one can see that it changes every time r changes). This might imply congestion on the lean up-link channel. Furthermore, since messages are delayed on the channels, the sender might increase the degree later than it should, thus sending symbols with a non-optimal degree. This scenario might reduce the revealing probability. Thus, we examine the worst case scenario where no feedback messages reach the sender and it continues to send encodings with the same degree of $d = \sqrt{k+1}$. This is the worst case scenario since any feedback message would cause the sender to increase the encodings degree which will improve the revealing probability of the encodings. We will show that the performance in this case is reasonable. That is, we consider the

scenario where once $k - \sqrt{k+1}$ symbols have been revealed, all the encodings have the same degree $d = \sqrt{k+1}$. We prove that even in this worst case scenario, the expected number of encodings needed for successfully revealing the last $\sqrt{k+1}$ symbols is $O(\sqrt{k} \log k)$ (if no feedback symbols are lost, then the expected number of encodings needed is $O(\sqrt{k})$). This result may be compared with the $O(k \log k)$ symbols required if all the feedback messages had been lost, and the sender would have kept sending with the same degree of $d = 1$.

Lemma 4.16: For decoding the last $\lfloor \sqrt{k+1} \rfloor$ symbols, if the sender encodes the symbols using the same degree of $d = \lfloor \sqrt{k+1} \rfloor$, then the expected number of symbols required to successfully decode these symbols is $O(\sqrt{k} \log k)$.

Proof: The expected number of encodings needed for revealing all the symbols in the range $r \in \{k - \lfloor \sqrt{k+1} \rfloor, \dots, k-1\}$ is $\sum_{r=k-\lfloor \sqrt{k+1} \rfloor}^{k-1} \frac{1}{P(d=\lfloor \sqrt{k+1} \rfloor, r)}$, which can be bounded by

$$\begin{aligned} & \sum_{r=k-\lfloor \sqrt{k+1} \rfloor}^{k-1} \frac{k}{d(k-r)} \prod_{j=1}^{d-1} \frac{k-j}{r+1-j} \\ &= \frac{k}{d} \sum_{i=1}^{\lfloor \sqrt{k+1} \rfloor} \frac{1}{i} \prod_{j=1}^{d-1} \frac{k-j}{(k-i)+1-j} \\ &\leq \frac{k}{\lfloor \sqrt{k+1} \rfloor} \left(\sum_{i=1}^{\lfloor \sqrt{k+1} \rfloor} \frac{1}{i} \right) \left(\frac{k}{k-2\lfloor \sqrt{k+1} \rfloor} \right)^{d-1} \\ &\leq 4\sqrt{k} \ln \sqrt{k} \cdot \left(1 + \frac{2\lfloor \sqrt{k+1} \rfloor}{k-2\lfloor \sqrt{k+1} \rfloor} \right)^{\lfloor \sqrt{k+1} \rfloor - 1} \\ &\leq 4\sqrt{k} \ln \sqrt{k} \cdot \left(1 + \frac{4}{\sqrt{k+1}} \right)^{\sqrt{k+1}} \\ &\leq 4\sqrt{k} \ln \sqrt{k} \cdot e^4 = O(\sqrt{k} \log k). \end{aligned}$$

■ This result shows that in the most problematic case, our scheme still copes with an entire feedback channel blocking while decoding the last $\sqrt{k+1}$ symbols and the efficiency of the process remains reasonable. However, if the feedback channel deteriorates throughout the decoding of the entire k symbols, the efficiency of the process (in terms of the number of symbols required for decoding) is reduced. In Section VI we will provide heuristic methods that improve the efficiency of the process despite some feedback channel faults.

V. OTHER APPLICATIONS

The data transfer protocol we have presented helps transferring a large amount of data between two thin parties over an extremely lossy channel, such as cellular or militarily environments. In this section we will show other applications for which our scheme is suitable. These applications are extension of previous works, e.g., [7], [11], [12], [20], [21].

A. Broadcast/Multicast Tree

Erasur correcting codes and exclusive-or based schemes fit broadcast and multicast through networks (whether they are cable, peer-to-peer, or satellites networks). We suggest extending our basic real-time oblivious schemes to the case of a (possible overlay) spanning broadcast tree. The spanning tree is a directed tree rooted at the sender and each node of the tree represents a receiver (one may restrict the receivers to be represented only by leaves). Each node of the tree takes an active role in the process, rather than just merely forwarding messages to its children. This is similar to the way network coding schemes are designed [17].

The nodes in the broadcast tree represent three types of processes: A server (or the root node), middle tier nodes, and leaves. The edges represent (possibly virtual) channels on which the transferring protocol is conducted between two peers. Each node may have several immediate children to which it transmits the data, and a single father from which it acquires information (with the exception of the root). Each channel might have different characteristics, in particular, a different loss pattern of packets.

Notice that the simple solutions of using the middle-tiers as relays of messages to their children will result in inefficient schemes. In one possible solution, the middle tiers simply act as routers while the server encodes messages to each receiver separately. This solution is not scalable and may increase traffic congestion on the server's links. In another solution, the middle tiers relay messages to the receivers, collect the minimal number of revealed symbols from each receiver, and send this information back to the sender. The server sends encodings of the minimal degree to all the receivers. This process reduces the revealing probability significantly for receivers with a large number of revealed symbols.

In our scheme, each non-leaf node u uses an independent connection session with each of its immediate children executing our basic real-time oblivious scheme. In this process, u receives from each child v the number of revealed symbols r_v that v has acquired. The sender u should generate encodings although it has revealed only r_u symbols of the k original symbols. To overcome this problem, u generates an encoding with degree $d(r = r_v, k = r_u)$ whose generating symbols are from the revealed symbols of u .

To analyze the performance of the tree scheme, it should be observed that in our RT-oblivious protocol the message size can be increased during the transferring process, while the revealing probability remains at least $1/e$. If both the sender and the receiver agree on the current message size k and the sender chooses $d(r, k)$ random symbols indices then the analysis of Lemma 4.3 remains valid and, by Lemma 4.8, the revealing probability is at least $1/e$. Thus, in this data broadcast scheme, each encoding reveals an input symbol with probability of at least $1/e$, and the expected number of symbols needed to reveal the message is at most ek .

However, in the worst case scenario, the complexity of the receiver might become $O(k^2)$, as encodings of high degree might be generated by the sender. This scenario is likely to take place if a middle-tier receives symbols in a rate which is

lower than a receiver's receiving rate.

B. Parallel Download and Migration

An additional application of our protocol is for downloading information from multiple servers. A single receiver may utilize the combined bandwidth of several servers for speeding up the download process when dealing with an extremely lossy communication channels. This can assist, for example, downloading multimedia files on a peer-to-peer network with peers using cellular communication.

A single receiver may contact several senders holding the entire message. The receiver conducts our RT-oblivious erasure scheme with each of these servers. The receiver informs each of the senders of the number of revealed symbols r after each decoding of a symbol (or after change of $d(r)$ with the enhancement of Lemma 4.14). Each sender will independently generate encodings according to the value of r it has last received from the receiver.

In the optimistic case where all symbols are generated using the optimal degree for the time the encoding is processed, the efficiency of the process is as seen before. Yet, if several servers simultaneously send the encoding with optimal degree for the current revealing state of the receiver, the efficiency of the process will slightly deteriorate as encoding of inappropriate degree might reach the receiver. However, this approach is still better than using a single server, as the number of symbols received by the combined bandwidth of all channels is much higher than the reduction of the revealing probability.

The extension of our scheme for downloading from several servers may be used for overcoming crashes of servers and for enabling addition of new download servers dynamically. This is achieved using the ratelessness property of our scheme, where encodings are independent, thus, allowing to decode the entire message from any appropriate set of encodings. An important example is using our scheme for migration between servers or relay stations on the fly. In order to add another server to download the message from, the receiver only needs to send the current number of symbols r it has revealed.

VI. HEURISTIC ENHANCEMENTS

As seen in the previous sections, our protocol does not achieve the best factor (namely, our protocol achieves a factor slightly less than 2) on the expected number of encoding symbols required to decode the k original symbols. We will now demonstrate heuristic methods for improving the efficiency of the protocol in common scenarios.

A. The First k Encodings

We send the k original symbols as the first k encodings, after which the original scheme follows. To analyze this heuristic, we consider the channel rate, denoted R , i.e., the probability that an encoding sent on the channel reaches the receiver. For a stable channel rate R , the expected number of symbols that reach the receiver is kR out of the original k symbols.

This improvement cannot diminish the efficiency of the protocol, and with higher channel rates, the efficiency improves.

The expected number of symbols that reach the receiver from the k encodings is kR . As for the rest of the $k - kR$ symbols being transmitted, by Lemma 4.8, the expected revealing probability is higher than $1/e$. Thus, the expected number of encodings required to reach the receiver for a successful decoding is at most $kR + k(1 - R)e = k(e - R(e - 1))$. For example, for $R = 0.95$ the expected number of symbols is $1.09k$.

As can be seen, with this enhancement of our scheme, as the channel rate R increases the expected number of symbols required for decoding is reduced, thus increasing efficiency. In this heuristic, we use the channel rate only for the analysis while the parties are not aware of the channel rate.

B. Decoding Rate Estimation

In our protocol the sender needs to know the number of symbols r that the receiver has revealed. We now address the cases in which the up-link channel delays and loses messages that the receiver sends containing the number of revealed symbols r . This information might not be relevant when needed, that is, when the encoded symbol reaches the receiver. If an encoding reaches the receiver with a non optimal degree, the revealing probability of that symbol decreases. There are different techniques through which the sender may approximate a more realistic value of r for the time the symbols reach the receiver. We next describe a technique for estimating this value of r .

We denote by R the channel rate, by W the channel latency, i.e., the time it takes a symbol to travel through the down or up link, and by T the time interval between two consequent encodings transmissions by the sender. The server may estimate the number of symbols the receiver has revealed by accumulating the probabilities by the non-discrete variable \hat{r} as shown in Fig. 5. This value would provide a good estimation for the expected number of symbols revealed by the receiver. The sender estimates the number of symbols revealed and not yet acknowledged as $2WR/eT$. It updates the current estimation of r accordingly, as explained below. The probability that a symbol sent by the sender reveals a symbol at the receiver is the probability that it reaches the receiver and reveals a symbol, that is, the probability is at least R/e . Thus, every time the sender sends a symbol, it adds R/e to \hat{r} . Furthermore, the number of symbols currently in the channel is W/T . Thus, when a feedback message arrives at the sender, it is a reply to a message sent $2W$ time units earlier, and since then $2W/T$ messages were sent by the sender. Hence, the expected number of symbols that are revealed by the server in this period is $2WR/eT$. Therefore, on a reception of a feedback message r' by the sender, the sender should adjust the value of \hat{r} to $r' + 2WR/eT$. In the protocol we assume that the sender knows W and R . We next explain how the estimation of the values W and R is done assuming an unstable channel.

In modern networks, the channel properties are highly unstable. Either packets are directed on different routes or the physical channel suffers from transient faults. Either way, the channel transition rate and the channel latency vary in

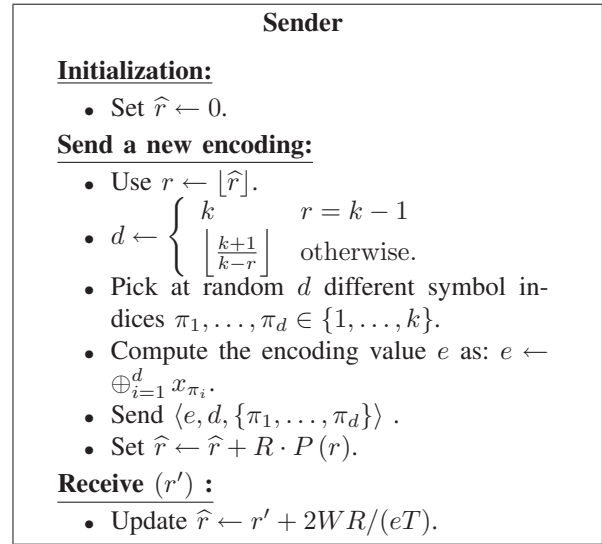


Fig. 5. The sender's protocol for estimating r assuming known stable channel properties.

time. It is necessary to have a mechanism for estimating the momentary channel properties.

A possible solution for estimating W is that the sender sends a sequence number with each encoding. The receiver, when replying to the sender, adds to the replied message the sequence number of the encoding that caused the revealing of a symbol. The sender, upon receiving the replied sequence number, may calculate the amount of time that passed between the encoding and the reply. Let Δ_i be the difference in sequence numbers between the current sequence number and the received sequence number. The channel latency may be estimated as $W = T\Delta_i/2$, as it is the time it takes for Δ_i symbols to travel back and forth.

To estimate the channel rate R , the receiver picks a window of a predetermined size. The receiver calculates the value of R in the window using the symbols sequence numbers. That is, it computes the ratio between the number of messages received in the window and the maximum difference in the sequence numbers in the window (possibly after removing outliers). The receiver sends the estimated rate with each replied message, that is, when $d(r)$ changes.

C. Sending the Indices of the Generating Symbols

In the RT oblivious protocol, as described in Section III, when an encoding generated from symbols with indices π_1, \dots, π_d is sent, the list of indices is sent with the encoding. This technique is inefficient for a message with a large degree, where the overhead is $d \log k$ bits to describe the indices. To overcome this problem, the sender can send a seed for a pseudo-random generator, enabling both the sender and the receiver to agree on a subset of indices of size d . Every encoding would consist of the degree d of the encoding, a seed for the pseudorandom generator used to define the indices of the generating symbols, and the encoding value (the bitwise exclusive-or value of all the generating symbols). Using a seed reduces the communication overhead, in comparison to sending the explicit set of indices.

We note that our analysis in Section IV is based on the assumption that the set of d indices is chosen with uniform distribution. The pseudo-random generator we use should be chosen such that the revealing probability is high.

REFERENCES

- [1] "The ATM forum," <http://www.atmforum.com>.
- [2] "Gigabit ethernet alliance," <http://www.gigabit-ethernet.org>.
- [3] A. S. Tanenbaum, *Computer Networks*, 4th ed. Prentice Hall, 2000.
- [4] J. Blömer, M. Kalfane, M. Karpinski, R. Karp, M. G. Luby, and D. Zuckerman, "An XOR-based erasure-resilient coding scheme," ICSI, Tech. Rep. 95-048, 1995.
- [5] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc. of the 29th Annu. ACM Symp. on the Theory of Computing*, 1997, pp. 150–159.
- [6] M. G. Luby, M. Mitzenmacher, and M. A. Shokrollahi, "Analysis of random processes via and-or tree evaluation," in *Proc. of the 9th Annu. ACM-SIAM Symp. on Discrete Algorithms*, 1998, pp. 364–373.
- [7] J. W. Byers, M. G. Luby, and M. Mitzenmacher, "Accessing multiple mirror sites in parallel: Using Tornado codes to speed up downloads," in *INFOCOM*, 1999, pp. 275–283.
- [8] S. Dolev, B. Fitingof, A. Melkman, and O. Tubman, "Smooth and adaptive forward erasure correcting," *Computer Networks*, vol. 36, pp. 343–355, 2001.
- [9] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 569–584, 2001.
- [10] M. G. Luby, "LT codes," in *Proc. of the 43rd Annu. IEEE Symp. on Foundations of Computer Science*, 2002, pp. 271–282.
- [11] P. Maymounkov and D. Mazières, "Rateless codes and big downloads," in *the 2nd International Workshop on Peer-to-Peer Systems (IPTPS03)*, 2003.
- [12] J. W. Byers, M. G. Luby, M. Mitzenmacher, and A. Rege, "Digital fountain approach to reliable distribution of bulk data," in *SIGCOMM*, 1998, pp. 56–67.
- [13] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. SIAM*, vol. 8, pp. 300–304, 1960.
- [14] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551 – 2567, 2006.
- [15] P. Maymounkov, "Online codes," NYU, Tech. Rep. TR2002-833, 2002.
- [16] D. S. Lun, M. Medard, and M. Effros., "On coding for reliable communication over packet networks," in *Proc. of the 42nd Annual Allerton Conference on Communication, Control and Computing*, 2004.
- [17] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.
- [18] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics, A foundation for Computer Science*, 2nd ed. Addison-Westley, 1998.
- [19] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, 1995.
- [20] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Computer Communication Review*, vol. 27, no. 2, pp. 24–36, 1997.
- [21] L. Rizzo and L. Vicisano, "A reliable multicast data distribution protocol based on software FEC techniques," in *the Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97)*, Sani Beach, Chalkidiki, Greece, June 1997.



Amos Beimel received the B.A., M.Sc., and D.Sc. degrees in Computer Science from the Technion – Israel Institute of Technology, Haifa, in 1989, 1992, and 1996, respectively. After graduating from the Technion he spent one year as a Postdoctoral Fellow at the Center for Discrete Mathematics and Computer Science (DIMACS) at Rutgers University, and two years as a Postdoctoral Fellow at the Division of Engineering and Applied Science at Harvard University. In 1999 he joined the Department of Computer Science at Ben-Gurion University, where he is now a senior lecturer. In 2005-2006, he spent a year as a visiting assistant professor at the University of California, Davis. His research interests include cryptography (mainly, private information retrieval protocols, secret sharing schemes, and secure computations) and complexity theory.



Shlomi Dolev Shlomi Dolev received his B.Sc. in Engineering and B.A. in Computer Science in 1984 and 1985, and his M.Sc. and D.Sc. in computer Science in 1990 and 1992 from the Technion Israel Institute of Technology. From 1992 to 1995 he was at Texas A&M University as a visiting research specialist. In 1995 he joined the Department of Mathematics and Computer Science at Ben-Gurion University where he is now a professor. He was a visiting researcher/professor at MIT, DIMACS, and LRI, for several periods during summers. Shlomi is the author of the book "self-stabilization" published by the MIT Press. He published more than hundred journal and conference scientific articles, and served in the program committee of more than thirty conferences including: the ACM Symposium on Principles of Distributed Computing, and the International Symposium on Distributed Computing. He is an associate editor of the AIAA Journal of Aerospace Computing, Information, and Communication. His research grants include IBM faculty awards, Intel academic grants, and the NSF. Shlomi established the computer science department at Ben-Gurion university, and served as the first chair of the department, where he now holds the Rita Altura trust chair in computer sciences. His current research interests include distributed computing, distributed systems, and communication networks; in particular the self-stabilization property of such systems.



Noam Singer received his B.A. and M.Sc. degrees in Computer Science from Ben-Gurion University, Beer-Sheva, Israel, in 1998, and 2005, respectively.

Between his B.Sc. and M.Sc., Noam worked in various communication companies developing ADSL and cable infrastructures. Noam joined Cisco Networks and is currently working on the Cisco Access Control Server project (Cisco ACS). His research interests include networking and cryptography.