Random Linear Network Coding For Time Division Duplexing: When To Stop Talking And Start Listening

Daniel E. Lucani
RLE, MIT
Cambridge, Massachusetts, 02139
Email: dlucani@mit.edu

Milica Stojanovic
Northeastern University
Boston, Massachusetts, 02118
Email: millitsa@mit.edu

Muriel Médard
RLE, MIT
Cambridge, Massachusetts, 02139
Email: medard@mit.edu

***Abstract*— A new random linear network coding scheme for reliable communications for time division duplexing channels is proposed. The setup assumes a packet erasure channel and that nodes cannot transmit and receive information simultaneously. The sender transmits coded data packets back-to-back before stopping to wait for the receiver to acknowledge (ACK) the number of degrees of freedom, if any, that are required to decode correctly the information. We provide an analysis of this problem to show that there is an optimal number of coded data packets, in terms of mean completion time, to be sent before stopping to listen. This number depends on the latency, probabilities of packet erasure and ACK erasure, and the number of degrees of freedom that the receiver requires to decode the data. This scheme is optimal in terms of the mean time to complete the transmission of a fixed number of data packets. We show that its performance is very close to that of a full duplex system, while transmitting a different number of coded packets can cause large degradation in performance, especially if latency is high. Also, we study the throughput performance of our scheme and compare it to existing half-duplex Go-back-N and Selective Repeat ARQ schemes. Numerical results, obtained for different latencies, show that our scheme has similar performance to the Selective Repeat in most cases and considerable performance gain when latency and packet error probability is high.**

I. Introduction

Network coding was introduced by Ahlswede *et al* [1]. This concept is also known as coded packet networks. Network coding considers the nodes to have a set of functions that operate upon received or generated data packets. Today's networks would represent a subset of the coded packet networks, in which each node has two main functions: forwarding and replicating a packet. A classical network's task is to transport packets provided by the source nodes unmodified. In contrast, network coding considers information as an algebraic entity, on which one can operate.

Network coding research originally studied throughput performance without delay considerations for the transmitted information. The seminal work by Ahlswede *et al* [1] considers a channel with no erasures and, therefore, no need for feedback. Work in [2] and [3] showed that linear codes over a network are sufficient to implement any feasible multicast connection, again considering a channel with no erasures. Also, [3] provides an algebraic framework for studying this subset of coded networks. In both of these cases, the nodes are considered to transmit a linear combination of the packets previously received. Work in [4] presents the idea of using linear codes generated randomly in a network and shows that it achieves multicast capacity in a non-erasure channel.

For networks with packet erasures, two approaches have been used. The first approach relies on rateless codes, i.e. transmitting coded data packets until the receiver sends an acknowledgement stating that all data packets have been decoded successfully. Reference [5] studies random linear network coding in lossy networks showing that it can achieve packet-level capacity for both single unicast and single multicast connections in wireline and wireless networks. Reference [6] presents network codes that preserve the communication efficiency of a random linear code, while achieving better computational efficiency. Reference [7] presented a random linear coding scheme for packet streams considering nodes with a fixed, finite memory, establishing a trade-off between memory usage and achievable rate. In terms of practical issues and implementation, work in [8] presents MORE, a MAC-independent opportunistic protocol for wireless networks, and provides experimental results with some emphasis on the throughput gains provided by network coding.

The work in [9] and [10] has studied delay performance gains and their scaling laws for network coding with and without channel side information, respectively. They focus on transmission of large files in a rateless fashion. In [11] the delay performance of network coding for a tree-based multicast problem is studied and compared to various Automatic Repeat reQuest (ARQ) and Forward Error Correcting (FEC) techniques. For network coding, it assumes reliable and instantaneous feedback to acknowledge a correct decoding of all data packets. Note that the focus of these references has been on either throughput or delay performance, usually considering minimal feedback.

Finally, the work in [12] couples the benefit of network coding and ARQ by acknowledging degrees of freedom (dof), defined as linearly independent combinations of the data packets, instead of original data packets to show that queue size in a node follows degrees of freedom.

The second approach uses block transmissions. Reference [13] studies the problem in wireless networks and shows that linear codes achieve capacity in the network. In [14] a queueing model for random linear coding is presented, which codes data packets in a block-by-block fashion using acknowledgements to indicate successful transmission of each block. Interestingly, the coding block size depends on the number of packets available in the queue, up to some maximum block size.

We study channels in which time division duplexing is necessary, i.e. when a node can only transmit or receive, but not both at the same time. This problem has not been considered in any of the previous network coding references or, to the best of our knowledge, for network coding before our work. This type of channel is usually called half-duplex in the literature, but we use the term time division duplexing (TDD) to emphasize that the transmitter and receiver do not use the channel half of the time each or in any pre-determined fashion. Important examples of time division duplexing channels are infrared devices (IrDA), which have motivated many TDD ARQ schemes [15] [16], and underwater acoustic communications [17]. Other important applications may be found in channels with very high latency, e.g. in satellite [18] [19], and deep space [20] communications.

More specifically, we focus on the problem of transmitting M data packets through a link using random linear network coding. We consider that the sender can transmit random linear coded packets back-to-back before stopping to wait for an acknowledgement (ACK) packet. This ACK packet conveys the remaining dofs required at the receiver to decode all M data packets. We consider that the number of coded packets N_i to be transmitted before waiting for a new ACK packet depends on the number of dofs i needed at the receiver, as indicated by the last ACK packet received successfully. If it is the first transmission, we consider that the required dofs is M. Figure 1 illustrates the communication process. The system transmits N_i coded packets (CP), and waits to receive an ACK packet that updates the value of i to j, at which point it will transmit N_j coded packets. The system will keep transmitting and stopping to update i, until $i = 0$. When $i = 0$, the transmitter can start with M new data packets, or simply stop. In Figure 1, $CP(k, d)$ represents the k-th coded packet transmitted when we start transmission with d dofs needed at the receiver to decode the information.

There is a natural trade off in the choice of the N_i's. Every time the system stops to wait for an ACK, it incurs in an additional delay, which can be large in high latency channels. In general, the system requires at least one stop to get confirmation of complete transmission. However, we want to minimize the number of stops required to complete transmission of the M packets. Note that if the N_i's are too small given the channel conditions, the system will have to transmit more ACK packets to complete transmission of the block of M data packets, which will cause a larger delay. On the other hand, if the N_i's are too large, the receiver will have decoded the M packets, for example, before the transmitter

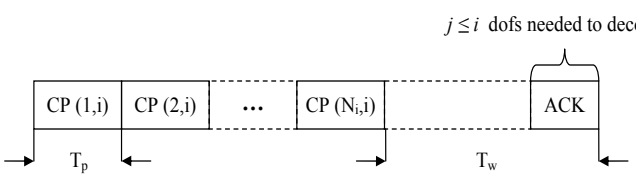

Fig. 1. Network coding TDD scheme

stops sending the first N_M coded packets. Since the block of M original packets is considered to be completely transmitted when the ACK requests no more dofs, the system causes unnecessary delay by transmitting too many coded packets by delaying transmission of the ACK by the receiver. In other words, the transmitter could have sent a smaller number of coded packets before stopping and still transmit the M packets successfully.

We show that there exists an optimal number of coded data packets to be transmitted back-to-back before stopping to wait for an ACK packet from the receiver, in terms of mean completion time, i.e. mean time to decode the M original data packets at the receiver and get an ACK at the transmitter. In fact, the optimal number of coded data packets N_i depends on the number of dofs i that the receiver requires to decode the information, and also on the packet error probability and the latency, i.e. the number of bits in flight. Thus, we show that there is an optimal time for stop transmitting coded packets and start listening to an ACK packet from the receiver.

Thus, our objective is to minimize the expected time to complete transmission of a block, i.e. the delay in block transmissions, using feedback. This delay to decode a block is different from the usual packet delay measure. Since coding is carried out on blocks of packets, the delay to decode a block successfully determines the delay of each of the packets in that block. We also show that minimizing the expected time to complete transmission of a block of M packets with a fixed packet size also maximizes the throughput performance. However, we show that a correct choice of M and number of bits in the data packet can further improve throughput performance.

Although both standard ARQ techniques and our scheme achieve reliability by detecting errors in received packets or packet erasures, and recover the information using a retransmission scheme, there are some important differences. First, we rely on transmission of coded packets, i.e. there is no need to specify a particular data packet to retransmit as in ARQ, but only a random linear combination. The ACK packet of our scheme thus differs from common ARQ techniques [21] in that it does not give acknowledgement to particular data packets [21], but to degrees of freedom needed at the receiver to decode the M original packets. Second, the number of coded packets transmitted in our scheme is not fixed by design of the algorithm, but chosen given channel characteristics and information in the ACK packet. In fact, the information in the ACK packet of our algorithm can be used to update an

estimate of the probability of packet error and improve the overall performance.

We present an analysis and numerical results that show that transmitting the optimal number of coded data packets sent before stopping to listen for an ACK provides performance very close to that of a network coding scheme operating in a full duplex channel, in terms of mean time to complete transmission of all packets. This is the case even in high latency channels. Choosing a number different from the optimum can cause a large degradation in performance, especially if latency is high.

Since random linear network coding is used, the results of this paper can be extended to the case of a network, in which each node performs a random linear combination of packets received from different nodes. In this extension, each node transmitting through a link, or, more generally, a hyperarc (using the terminology in [22]) will have an optimal number of coded packets to transmit back-to-back before stopping to listen.

The paper is organized as follows. In Section 2, we present the setup of the problem. In Section 3, we present the analysis of expected time to complete transmission of M data packets and the optimization required to determine the number of coded packets to transmit before stopping. Also, two network coding comparison schemes are presented. In Section 4, the throughput performance is analyzed. In Section 5, numerical results are presented comparing our TDD optimal network coding with several other schemes in terms of the mean time to complete transmission. We also present results that compare throughput performance of our scheme to that of typical ARQ schemes. Conclusions are summarized in the Section 6.

II. Random Network Coding for TDD channels

A sender in a link wants to transmit M data packets at a given link data rate R. The channel is modeled as a packet erasure channel. Nodes can only transmit or receive, but not both at the same time. The sender uses random linear network coding [4] to generate coded data packets. Each coded data packet contains a linear combination of the M data packets of n bits each, as well as the random encoding coefficients used in the linear combination. Each coefficient is represented by g bits. For encoding over a field size q, we have that $g = \log_2 q$ bits. Also consider an information header of size h. Thus, the total number of bits per packet is $h+n+gM$. Figure 2 shows the structure of each coded packet consider in our scheme.

The sender can transmit coded packets back-to-back before stopping to wait for the ACK packet. The ACK packet feeds back the number of degrees of freedom, that are still required to decode successfully the M data packets. Since random linear coding is used, there is some probability of choosing encoding vectors that are all zero for one coded packet or encoding vectors that are linearly dependent on vectors of previously received packets. Thus, using arguments similar to [9], the expected number of successfully received packets

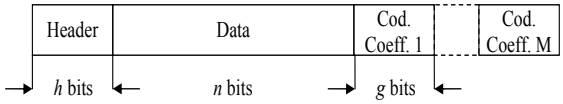

Fig. 2. Structure of coded data packet

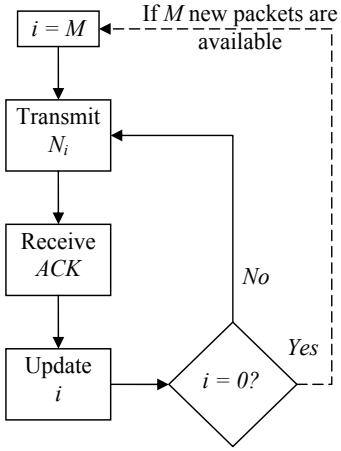

Fig. 3. Algorithm of network coding for time division duplexing channels. i represents the remaining number of degrees of freedom to decode the packets, and N_i the corresponding number of coded packets transmitted before stopping to listen for a new ACK. The ACK packet has the information to update i

before having M linearly independent combinations, is

$$\sum_{k=1}^{M} \frac{1}{(1-(1/q)^k)} \leq M \frac{q}{q-1} \tag{1}$$

In the following analysis, we assume that the field size q is large enough so that the expected number of successfully received packets at the receiver, in order to decode the original data packets, is approximately M. This is not a necessary assumption for our analysis. We could have included the probabilities of receiving linearly independent combinations into the transition probabilities. However, making this assumption simplifies the expressions and provides a good approximation for large enough q.

We are interested in determining the optimal number of coded packets that should be sent back-to-back before waiting for an ACK packet from the receiver in order to minimize the time for successfully transmitting the M data packets over the link.

Note that if M packets are in the queue, at least M degrees of freedom have to be sent in the initial transmission, i.e. $N_M \geq M$ coded packets. We are interested not only in the number of dof that are required at the first transmission, but also at subsequent stages. Transmission begins with M information packets, which are encoded into N_M random linear coded packets and transmitted. If all M packets are decoded successfully, the process is completed. Otherwise, the ACK informs the transmitter how many are missing, say

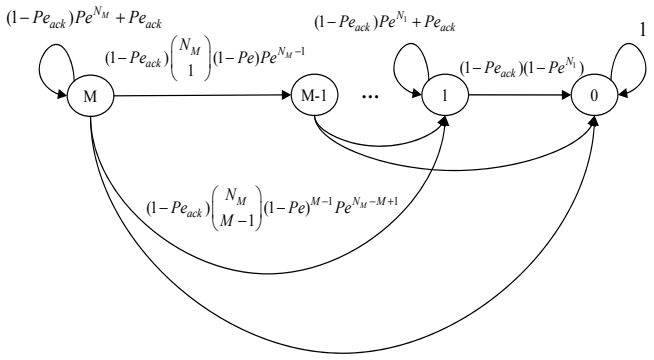

Fig. 4. Markov chain representation of the scheme. State i represents that the receiver requires i more successfully received coded packets to decode the information

i. The transmitter then sends N_i coded packets, and so on, until all M packets have been decoded successfully. We are interested in the optimal number N_i of coded packets to be transmitted back-to-back in the next transmission to complete the remaining i dof's. Figure 3 shows the communication process as a system transmits N_M coded packets initially and awaits reception of an ACK packet that updates the value of i, at which point it will transmit N_i coded packets. The system will keep transmitting and stopping to update i, until i = 0. When i = 0, the transmitter can start with M new data packets or simply stop. In Figure 1, $CP(k,d)$ represents the k-th coded packet transmitted when we start transmission with d dofs needed at the receiver to decode the information.

The process can be modelled as a Markov Chain (Figure 4). The states are defined as the number of dof's required at the receiver to decode successfully the M packets. Thus, these states range from M to 0. This is a Markov Chain with M transient states and one recurrent state (state 0). Let us define N_i as the number of coded packets that are sent when i dof's are required at the receiver in order to decode the information. Note that the time spent in each state depends on the state itself, because $N_i \neq N_j, \forall i \neq j$ in general.

The transition probabilities from state i to state j ($P_{i\to j}$) have the following expression for $0 < j < i$ and $N_i \geq i$:

$$P_{i\to j} = (1-Pe_{ack})\binom{N_i}{i-j}(1-Pe)^{i-j}Pe^{N_i-i+j} \tag{2}$$

where Pe and Pe_{ack} represents the erasure probability of a coded packet and of an ACK packet, respectively.

More generally, the transition probability can be defined for any value of $N_i \geq 1$ as follows:

$$P_{i\to j} = (1-Pe_{ack})f(i,j)(1-Pe)^{i-j}Pe^{N_i-i+j} \tag{3}$$

where

$$f(i,j) = \begin{cases} \binom{N_i}{i-j} & \text{if } N_i \geq i, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

For $j = i$ the expression for the transition probability reduces to:

$$P_{i\to i} = (1-Pe_{ack})Pe^{N_i} + Pe_{ack} \tag{5}$$

III. Expected Time for completing transmission

The expected time for completing the transmission of the M data packets constitutes the expected time of absorption, i.e. the time to reach state 0 for the first time, given that the initial state is M. This can be expressed in terms of the expected time for completing the transmission given that the Markov Chain is in state is i, T_i , $\forall i = 0, 1, ..M-1$. Let us denote the transmission time of a coded packet as T_p, and the waiting time to receive an ACK packet as T_w. For our scheme, $T_p = \frac{h+n+gM}{R}$ and $T_w = T_{rt} + T_{ack}$, where $T_{ack} = n_{ack}/R$, n_{ack} is the number of bits in the ACK packet, R is the link data rate, and T_{rt} is the round trip time. Note that $T_0 = 0$. Then, for $i > 1$:

$$T_i = \frac{N_iT_p + T_w}{(1-Pe_{ack})(1-Pe^{N_i})} \tag{6}$$

$$+ \frac{(1-Pe)^i Pe^{N_i-i}\sum_{j=1}^{i-1} f(i,j)\left(\frac{Pe}{1-Pe}\right)^j T_j}{1-Pe^{N_i}} \tag{7}$$

For example, for i = 1 we have that:

$$T_1 = \frac{\left(N_1T_p + T_w\right)}{(1-Pe_{ack})(1-Pe^{N_1})} \tag{8}$$

As it can be seen, the expected time for each state i depends on all the expected times for the previous states. Because of the Markov property, we can optimize the values of all N_i's in a recursive fashion, i.e. starting by N_1, then N_2 and so on, until N_M, in order to minimize the expected transmission time. We do so in the following subsection.

A. Minimizing Expected Time for Completing Transmission

Our objective is to minimize the value of the expected transmission time T_M. Under the assumption that $N_i \geq i$, we have:

$$\min_{N_M,..,N_1} T_M = \min_{N_M,..,N_1} \frac{N_MT_p+T_w}{(1-Pe_{ack})(1-Pe^{N_M})} + \frac{(1-Pe)^M Pe^{N_M-M}\sum_{j=1}^{M-1}\binom{N_M}{M-j}\left(\frac{Pe}{1-Pe}\right)^j T_j}{1-Pe^{N_M}} \tag{9}$$

$$= \min_{N_M} \frac{N_MT_p+T_w}{(1-Pe_{ack})(1-Pe^{N_M})} + \frac{(1-Pe)^M Pe^{N_M-M}\sum_{j=1}^{M-1}\binom{N_M}{M-j}\left(\frac{Pe}{1-Pe}\right)^j \min_{N_j,..,N_1} T_j}{1-Pe^{N_M}} \tag{10}$$

Without this assumption, we have that

$$\min_{N_M,..,N_1} T_M =$$

$$\min_{N_M,..,N_1} \frac{N_M T_p + T_w}{(1-Pe_{ack})(1-Pe^{N_M})} \tag{11}$$

$$+\frac{(1-Pe)^M Pe^{N_M - M} \sum_{j=1}^{M-1} f(M,j)\left(\frac{Pe}{1-Pe}\right)^j T_j}{1-Pe^{N_M}}$$

$$= \min_{N_M} \frac{N_M T_p + T_w}{(1-Pe_{ack})(1-Pe^{N_M})} \tag{12}$$

$$+\frac{(1-Pe)^M Pe^{N_M - M} \sum_{j=1}^{M-1} f(M,j)\left(\frac{Pe}{1-Pe}\right)^j \min_{N_j,..,N_1} T_j}{1-Pe^{N_M}}$$

Hence, regardless of the assumption on N_i, the problem of minimizing T_M in terms of the variables $N_M, .., N_1$ can be solved iteratively. First, we compute $\min_{N_1} T_1$, then use this results in the computation of $\min_{N_2,N_1} T_2$, and so on.

One approach to computing the optimal values of N_i is to ignore the constraint to integer values and take the derivative of T_i with respect to N_i and look for the value that sets it equal to zero. For our particular problem, this approach leads to solutions without a closed form, i.e. expressed as an implicit function. For $M = 1$, the optimal value of N_1 can be expressed using a known implicit function (Lambert function), and it is given by

$$N_1^* = \frac{1 + W\left(-\exp\left(-1 + \frac{\ln(Pe) T_w}{T_p}\right)\right)}{\ln Pe} - \frac{T_w}{T_p} \tag{13}$$

where $W(\cdot)$ is the Lambert W function [23]. The positive values are found for the branch W_{-1}, as denoted in reference [23].

The case of $M = 1$ can be thought as an optimized version of the uncoded Stop-and-Wait ARQ, which is similar to the idea presented in [18]. Instead of transmitting one packet and waiting for the ACK, our analysis suggests that there is an optimal number of back-to-back repetitions of the same data packet that should be transmitted before stopping to listen for an ACK packet.

Instead of using the previous approach, we perform a search for the optimal values $N_i, \forall i \in \{1, ...M\}$, using integer values. Thus, the optimal N_i's can be computed numerically for given Pe, Pe_{ack}, T_w and T_p. In particular, the search method for the optimal value can be made much simpler by exploiting the recursive characteristic of the problem, i.e. instead of making a M-dimensional search, we can perform M one-dimensional searches. Finally, these N_i's do not need to be computed in real time. They can be pre-computed and store in the receiver as look-up tables. This procedure reduces the computational load on the nodes at the time of transmission.

B. Comparison Scheme 1: Fixed Maximum Window

Let us consider the same setting, i.e. a fixed number of packets M that have to be transmitted to the receiver, but with a fixed, pre-determined maximal number of coded packets to be transmitted before stopping to listen. We define this maximal value of coded packets as ω. If the number of degrees of freedom i required at the receiver to decode the information is $i \geq \omega$, the transmitter will transmit ω degrees of freedom. If $i < \omega$, the transmitter will transmit i degrees of freedom.

The model for the Markov Chain is derived from the previous case, by setting $N_i = \omega, \forall i \geq \omega$ and $N_i = i, \forall i < \omega$. For $i \geq \omega$, we have that:

$$T_i = \frac{\omega T_p + T_w}{(1-Pe_{ack})(1-Pe^{\omega})} \tag{14}$$

$$+\frac{\sum_{j=1}^{\omega} \binom{\omega}{j} \left(Pe^{\omega - j}(1-Pe)^j\right) T_{i-j}}{1-Pe^{\omega}} \tag{15}$$

and for $i < \omega$:

$$T_i = \frac{iT_p + T_w}{(1-Pe_{ack})(1-Pe^{i})} \tag{16}$$

$$+\frac{\sum_{j=1}^{i} \binom{i}{j} \left(Pe^{i - j}(1-Pe)^j\right) T_{i-j}}{1-Pe^{i}} \tag{17}$$

C. Comparison Scheme 2: Optimal Full Duplex ARQ

This scheme assumes that nodes are capable of receiving and transmitting information simultaneously, and in that sense it is optimal in light of minimal delay. The sender transmits coded packets back-to-back until an ACK packet for correct decoding of all information (M information packets) has been received. This scheme can be modeled as a Markov Chain where, as before, the states represent the number of dofs received. The time spent in each state is the same (T_p). Once the M packets have been decoded, i.e. M dofs have been received, the receiver transmits ACK packets back-to-back, each of duration T_{ack}. One ACK should suffice but this procedure minimizes the effect of a lost ACK packet.

The mean time to complete the transmission and get and ACK is:

$$E[T] = T_{rt} + \frac{MT_p}{1-Pe} + \frac{T_{ack}}{1-Pe_{ack}} \tag{18}$$

where T is the time to complete transmission of M packets.

IV. Throughput

The mean throughput is defined as $E[\frac{Mn}{T}]$, where T is the time to complete transmission of M packets. For Mn deterministic we have $MnE[\frac{1}{T}]$. For the case of $M = 1$, i.e. the extended version of the Stop-and-Wait ARQ scheme, we can provide a simple expression for the mean throughput in terms of the transition probabilities $P_{1\to1}$ and $P_{1\to0}$,

$$E[\frac{1}{T}] = \frac{P_{1\to0}}{P_{1\to1}} \sum_{k=1}^{\infty} \frac{{P_{1\to1}}^k}{k(T_p + T_w)} \tag{19}$$

$$= \frac{P_{1\to0}}{P_{1\to1}(T_p + T_w)} \sum_{k=1}^{\infty} \frac{(1-P_{1\to0})^k}{k} \tag{20}$$

$$= -\frac{P_{1\to0}}{P_{1\to1}(T_p + T_w)} \ln(P_{1\to0}) \tag{21}$$

We have used the Mercator series since $|1 - P_{1\to0}| < 1$ for all cases of interest. However, for $M > 1$ this expressions are complicated. Thus, we define our measure of throughput η as the ratio between number of data bits transmitted (N) and the

time it takes to transmit them. For the case of a block-by-block transmission, as described in Section III,

$$\eta = \frac{Mn}{T_M} \tag{22}$$

where T_M is the expected time of completion defined previously.

Note that the expected throughput and η are not equal. For the case of $M = 1$, note that $E[\frac{Mn}{T}] = \eta \frac{\ln(1/P_{1\to 0})}{P_{1\to 1}}$. More generally, using Jensen's inequality, $MnE[\frac{1}{T}] \geq \frac{Mn}{T_M}$ for $T > 0$. Therefore, η constitutes a lower bound to the mean throughput in our scheme. Another reason to consider this measure is to compare our network coding scheme with typical ARQ schemes that do not rely on coded packets since the analysis for most ARQ schemes is performed using η.

Note that if M and n are fixed, η is maximized as T_M is minimized. Thus, by minimizing the mean time to complete transmitting of a block of M data packets with n bits each, we are also maximizing η for those values. However, we show that the maximal η should be obtained using M and n as arguments in our optimization.

This is important for systems in which the data is streamed. In this case, searching for the optimal values of M and n, in terms of η, provides a way to optimally divide data into blocks of M packets with n bits each before starting communication using our scheme.

A. *Optimal Packet size and packets per block*

We have discussed throughput with a pre-determined choice of the number of data bits n and the number of data packets M in each block. However, expression 22 implies that the throughput η depends on both n and M. Hence, it is possible to choose these parameters so as to maximize the throughput. We can approach this problem is several ways. The first approach is to look for the optimal n while keeping M fixed:

$$\eta_{opt}(M) = \arg\max_{n} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{23}$$

The second approach is to look for the optimal M while keeping n fixed:

$$\eta_{opt}(n) = \arg\max_{M} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{24}$$

More generally, we could consider the case in which both parameters are variable and we are interested in maximizing η:

$$\eta_{opt} = \arg\max_{n,M} \left\{ \max_{N_M,\ldots,N_1} \eta \right\} \tag{25}$$

V. Numerical Examples

This section provides numerical examples that compare the performance of the different network coding schemes we have discussed so far in TDD channels. The comparison is carried out in terms of the mean time to complete a transmission of M data packets through TDD channel under different block error probabilities. We also present results in terms of the measure of throughput η to illustrate its dependence on the values of M and n for varying channel characteristics (erasure probabilities). We use the case of satellite communications as an example of high latency channels.

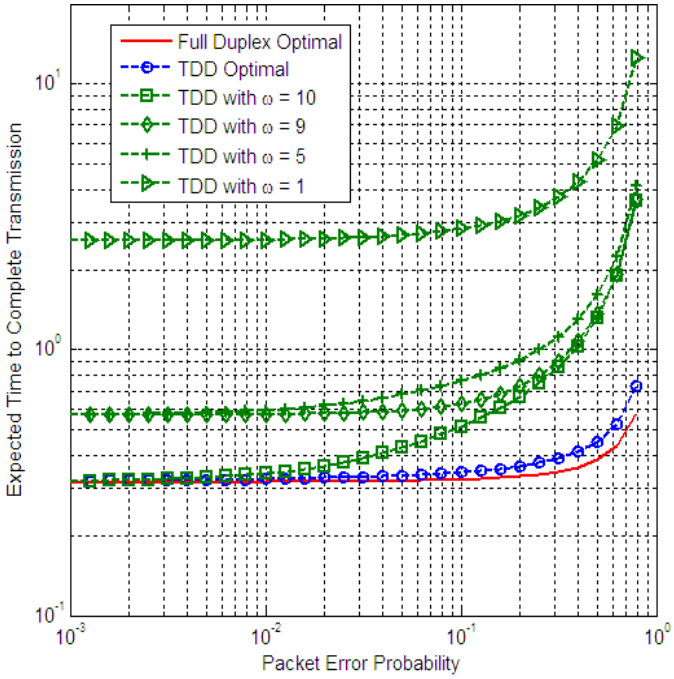

Fig. 5. Expected time for transmitting M data packets successfully versus Pe in a satellite example. The parameters used are $M = 10$, $T_{rt} = 250$ ms, data rate 1.5 Mbps, $n_{ack} = 100$ bits, $n = 10000$ bits, $g = 100$ bits, $h = 80$ bits, $Pe_{ack} = 0.001$

Figure 5 studies the expected time to complete transmission of $M = 10$ data packets of size $n = 10000$ bits, with different packet error probabilities in a GEO satellite link with a propagation delay of 125 ms. We assume a link with parameters specified in the figure. Note that our network coding scheme (TDD optimal) and the network coding full duplex optimal scheme have similar performance over a wide range of block error probabilities. In fact, for the worst case ($Pe = 0.8$) presented in this figure, our scheme has an expected time of completion only 29 % above the full duplex scheme. This is surprising considering that the transmitter in the full duplex scheme sends coded packets non-stop until an ACK packet is received. The explanation for this behavior is that our scheme is sending enough coded packets, given the channel conditions, so that the number of stops to listen (which are very costly) is minimized. Thus, our scheme can have similar performance to that of full duplex optimal scheme, in the sense of expected time to completion. Most importantly, our scheme is very likely to have a much better performance in terms of energy consumption due to the long periods in which the transmitter stops to listen for the ACK packets.

Figure 5 also shows the performance of the comparison scheme 1 presented in Section III. Note that when $\omega = 10$, i.e. the transmitter sends at most 10 coded packets before stopping to listen, the performance is comparable to our optimal scheme

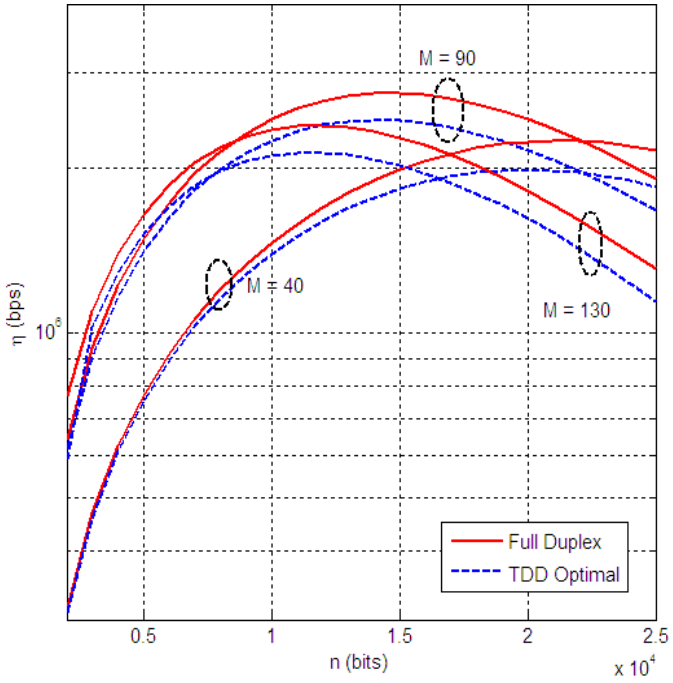

Fig. 6. Throughput measure η versus the number of bits n in a data packet for a symmetrical channel, for different values of M with parameters $g = 100$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 100 Mbps, $T_{rt} = 250$ ms, $Pe_{bit} = 0.0001$

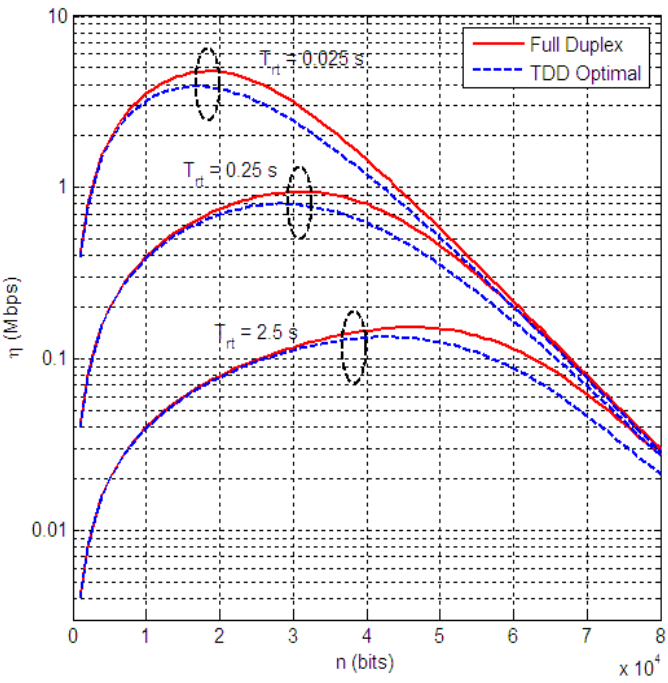

Fig. 7. Throughput η versus n in a symmetrical channel considering different values of round-trip time T_{rt} with parameters $g = 100$ bits, $n_{ack} = 100$ bits, $h = 80$ bits, data rate 1.5 Mbps, $M = 10$, $Pe_{bit} = 0.0001$

when the block error probability is low. This fact confirms that for low block error probabilities the optimal choice of coded packets to transmit when i dof are required at the receiver (N_i) is simply i. In other words, if $M = 10$ and the block error probability is low, the first transmission contains 10 coded packets. Note that using $\omega = 9$ already suffers from a considerable degradation in performance even for low Pe because the transmitter cannot transmit the minimum number of coded packets (M) necessary to decode the information after the first transmission, and so it must transmit at least one more coded packet after the first ACK. Note that the performance of $\omega = 5$ and $\omega = 9$ is similar for low block error probability because both of them require at least two stops to listen for ACK packets in order to relay all the information, and it is the stopping time that affects delay the most on a high latency channel. For the case of $\omega > 10$ we would see a degradation for low Pe, with respect to optimum, because more packets than necessary are transmitted.

Finally, note that for the worst data error probability in Figure 5, all fixed schemes (TDD with fixed ω) take at least 5 times more time to complete transmission than the network coding full duplex optimal scheme. The case of $\omega = 1$ can be interpreted as the performance of the Stop-and-Wait ARQ scheme under the same channel conditions, which is considerably worse than the other schemes.

Let us turn our attention now to the problem of maximizing the parameter η, i.e. our mean throughput lower bound. Recall that for this setting we are streaming data which is subdivided into blocks that are transmitted them using our scheme. Considering again a satellite link, given a fixed bit error probability ($Pe_{bit} = 0.0001$) let us study the problem of computing the optimal number of bits n per packet given some value of M. In these examples, for the case of a symmetric channel with independent bits $Pe = 1 - (1 - Pe_{bit})^{h+n+gM}$ and $Pe_{ack} = 1 - (1 - Pe_{bit})^{n_{ack}}$.

Figure 6 illustrates the values of η in Mbps given different choices of M and n. First, note that for each value of M there exists an optimal value of n. Thus, an arbitrary choice of n can produce a considerable degradation in performance in terms of throughput. Secondly, there is a (M, n) pair that maximizes the value of η. Finally, the performance of the full duplex network coding and our TDD optimal scheme is comparable for different values of n and M.

Figure 7 shows η in Mbps when we change the round-trip time T_{rt}. As expected, a lower T_{rt} allows more throughput in TDD. Again, we observe that our TDD optimal scheme has comparable performance to the full duplex scheme.

Let us compare the performance of our optimal TDD network coding scheme with respect to typical TDD ARQ schemes: Go-back-N (GBN) and Selective Repeat (SR). For this comparison, we use the η factor for the half-duplex version's of these schemes. Reference [15] studied both of these cases and proposed the utilization factor. In our notation, the equivalent η's are given by η_{GBN} and η_{SR} for GBN and SR, respectively:

$$\eta_{GBN} = \frac{n(1 - Pe)\left(1 - (1 - Pe)^W\right)}{(WT_p + T_w)Pe} \tag{26}$$

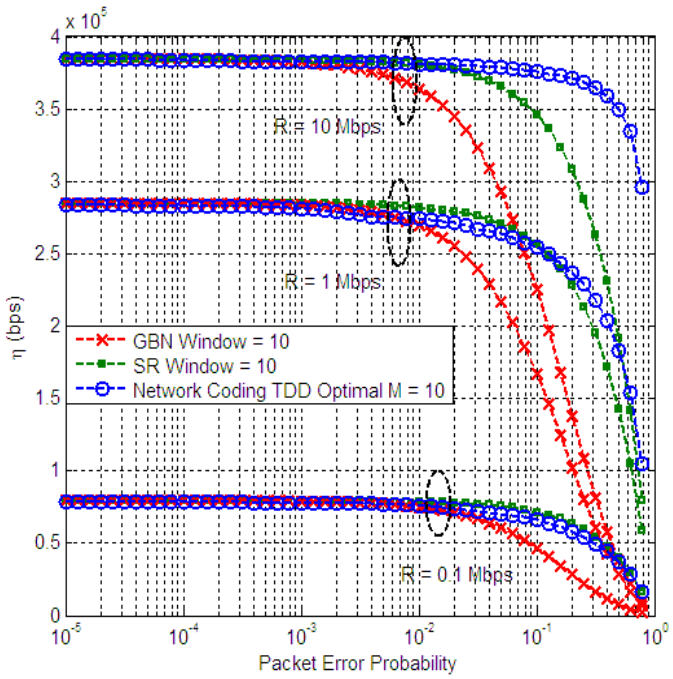

Fig. 8. η versus data packet error probability with two TDD non-network coding schemes (Go-Back-N and Selective Repeat) and our optimal TDD network coding scheme, with different R. We used as parameters $g = 20$ bits, $n_{ack} = 100$ bits, $n = 10000$ bits, $h = 80$ bits, $T_{rt} = 0.25$ ms

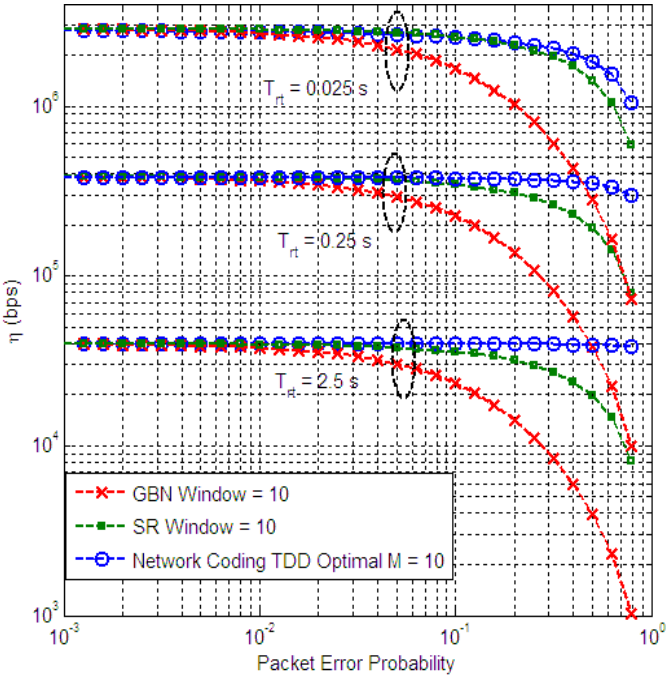

Fig. 9. η versus data packet error probability with two TDD non-network coding schemes (Go-Back-N and Selective Repeat) and our optimal TDD network coding scheme, with different T_{rt} values. We used as parameters $g = 20$ bits, $n_{ack} = 100$ bits, $n = 10000$ bits, $h = 80$ bits, $R = 10$ Mbps

and

$$\eta_{SR} = \frac{Wn(1-Pe)}{WT_p + T_w} \tag{27}$$

where W is the window size.

Figure 8 shows η for the satellite communications setting with a fixed packet size of $n = 10000$ bits, $n_{ack} = 100$ bits, $T_{rt} = 250$ ms, $Pe_{ACK} = 0$ for all schemes, a window size of $W = 10$ for the ARQ schemes, and $g = 20$ bits and $M = 10$ for our network coding scheme. We use different data rates to illustrate different latency scenarios, where higher data rate is related to higher latency. Note that the performance of our scheme is the same as both GBN and SR at low data packet error probability, which is expected because the window size W is equal to the block size of our scheme M and we expect very few errors. Our scheme has a slightly lower η for low Pe because each coded data packet includes gM additional bits that carry the random encoding vectors. This effect is less evident as latency increases. In general, our scheme has better performance than GBN.

Figure 8 shows that for low latency (0.1 Mbps) η of our scheme is very close to that of the SR ARQ scheme for all values of Pe, and better than the GBN scheme for high Pe. These results are surprising, because our scheme constitutes a block-by-block transmission scheme which will not start transmission of a new set of M data packets until the previous ones have been received and acknowledged. Note also that, as latency increases, our scheme shows much better performance than the SR scheme for high Pe. The case of 10 Mbps and $Pe = 0.8$ shows that η of our scheme is more than three (3) times greater than that of SR.

Figure 9 shows η for a fixed data rate of 10 Mbps and different T_{rt}. We use a fixed packet size of $n = 10000$ bits, $n_{ack} = 100$ bits, $Pe_{ACK} = 0$ for all schemes, a window size of $W = 10$ for the ARQ schemes, and $g = 20$ bits and $M = 10$ for our network coding scheme. Note that the overhead of transmitting M coefficients of g bits per coded packet is only 2%. Thus, this effect cannot be appreciated in the figures. Again, the performance of our scheme is the same as both GBN and SR at low data packet error probability. Since the data rate is kept fixed, at higher T_{rt} we get higher latency. The throughput performance is similar to that observed in Figure 8 if we carry our comparison in terms of latency.

Another advantage of our scheme with respect to SR ARQ is that our scheme relies on transmitting successfully one block of M data packets before transmitting a new one. In fact, our scheme minimizes the delay of every block. In contrast, the SR ARQ does not provide any guarantee of delay for any data packet, e.g. the first packet of a file to be transmitted could be the last one to be successfully received. In this sense, our comparison is not completely fair, as it favors the standard schemes. Nonetheless, our scheme is providing similar or better performance than SR but guaranteeing low transmission delays in individual data packets.

VI. Conclusion

This paper proposes a new random linear network coding scheme for reliable communications for time division

duplexing channels. This scheme optimizes the mean time to complete transmission of a number of data packets by determining the number of coded data packet that the sender has to transmit back-to-back before stopping to wait for the receiver to acknowledge how many degrees of freedom, if any, are required to decode correctly the information.

The optimal number of coded data packets, in terms of mean completion, to be sent back-to-back depends of the latency, probabilities of erasure of the coded packet and the ACK, and the number of degrees of freedom that the receiver requires to decode the data. While there is no closed form solution for the optimal number of packets, we can perform a search of the optimal values. In particular, the search method for the optimal value is simple by exploiting the recursive characteristic of the problem, i.e. instead of making a M-dimensional search, we perform M one-dimensional searches. Finally, these values do not need to be computed in real time. They can be pre-computed and stored in the receiver as look-up tables. This procedure makes the computational load on the nodes to be negligible at the time of determining the optimal transmission time.

We present means of analysis and numerical results to show that transmitting the optimal number of coded packets before stopping to listen for an ACK is very close to the performance of a full duplex system, while choosing a different number can cause considerable degradation in performance, especially if latency and packet error probability are high. Notably, our scheme also shows good potential to improve energy consumption.

In terms of throughput performance, we compare our scheme to the standard half-duplex Go-back-N and Selective Repeat ARQ schemes. Numerical evaluation for different latency shows that our scheme has similar performance to the Selective Repeat in most cases, and considerable performance gain when latencies and packet error probability are high. Numerical results also show that our scheme is superior to Go-back-N when error probability is high for different latency.

Future research will consider the problem of 1) energy consumption and associated optimization for this scheme, and 2) extension of the principles proposed (which were analyzed for one link in a network) to the general problem of wireless networks.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation under grants No. 0520075, 0427502, and CNS-0627021, by ONR MURI Grant No. N00014-07-1-0738, and subcontract # 060786 issued by BAE Systems National Security Solutions, Inc. and supported by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare System Center (SPAWARSYSCEN), San Diego under Contract No. N66001-06-C-2020 (CBMANET).

REFERENCES

[1] Ahlswede, R., Cai, N., Li, S. Y. R., Yeung, R. W., "Network Information Flow", IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204-1216, Jul. 2000

[2] Li, S.Y.R., Yeung, R. W., Cai, N., "Linear Network Coding", IEEE Trans. Inf. Theory, vol. 49, pp. 371, Feb. 2003

[3] Koetter, R., Médard, M., "An algebraic Approach to network coding", IEEE/ACM Trans. Netw., vol. 11, no. 5, pp. 782-795, Oct. 2003

[4] Ho, T., Medard, M., Koetter, R., Karger, D.R., Effros, M., Shi, J., Leong, B.,"A Random Linear Network Coding Approach to Multicast", Trans. Info. Theory, vol. 52, no. 10, pp.4413-4430, Oct. 2006

[5] Lun, D. S., M'edard, M., Koetter, R., Effros, M., "On coding for reliable communication over packet networks", Physical Comm., Vol. 1, Issue 1, pp. 3-20, Mar. 2008

[6] Maymounkov, P., Harvey, N. J. A., Lun, D. S., "Methods for Efficient Network Coding", In Proc. 44-th Allerton Conference'06

[7] Lun, D. S., Pakzad, P., Fragouli, C., M'edard, M., Koetter, R., "An Analysis of Finite-Memory Random Linear Coding on Packet Streams", In Proc. WiOpt'06, Boston, MA, USA, Apr. 2006

[8] Chachulski, S., Jennings, M., Katti, S., Katabi, D., "Trading Structure for Randomness in Wireless Opportunistic Routing", In Proc. COMM'07, pp. 169-180, Kyoto, Japan, Aug. 2007

[9] Eryilmaz, A., Ozdaglar, A., Médard, M., "On Delay Performance Gains from Network Coding", In Proc. CISS'06, pp. 864-870, Princeton, NJ, USA, Mar. 2006

[10] Ahmed, E., Eryilmaz, A., Médard, M., Ozdaglar, A., "On the Scaling Law of Network Coding Gains in Wireless Networks", In Proc. MILCOM'07, Orlando, FL, USA, Oct. 2007

[11] Ghaderi, M., Towsley, D., Kurose, J., "Network Coding Performance for Reliable Multicast", In Proc. MILCOM'07, Orlando, FL, USA, Oct. 2007

[12] Sundararajan, J. K., Shah, D., M'edard, M., "ARQ for Network Coding", In Proc. ISIT'08, pp. 1651-1655, Toronto, Canada, Jul. 2008

[13] Dana, A. F., Gowaikar, R., Palanki, R., Hassibi, B., Effros, M., "Capacity of wireless erasure networks", IEEE Trans. on Info. Theory, vol. 52, no. 3, pp. 789-804, Mar. 2006

[14] Shrader, B., Ephremides, A., "A queueing model for random linear coding", In Proc. MILCOM'07, Orlando, USA, Oct. 2007

[15] Ozugur, T., Naghshineh, M., Kermani, P., Copeland, J. A.,"On the performance of ARQ protocols in infrared networks", Int. Jour. Commun. Syst., vol. 13, pp. 617-638, 2000

[16] Shah, A. M., Ara, S. S., Matsumoto, M.,"An improved selective-repeat ARQ scheme for IrDA links at high bit error rate", In Proc. 4th int. conf. on Mobile and ubiquitous multimedia, Christchurch, New Zealand, pp. 37-42, 2005

[17] Stojanovic, M., "Optimization of a Data Link Protocol for an Underwater Acoustic Channel", In Proc. Oceans 2005 - Europe, pp. 68- 73, Jun. 2005

[18] Sastry, A. R. K., "Improving Automatic Repeat-Request (ARQ) Performance on Satellite Channels Under High Error Rate Conditions", IEEE Trans. on Comms., vol. 23, no. 4, pp. 436-439, Apr. 1975

[19] Yu, P., Lin, S., "An Efficient Selective-Repeat ARQ Scheme for Satellite Channels and Its Throughput Analysis", IEEE Trans. on Comms., vol. 29, no. 3, pp. 353-363, Mar. 1981

[20] Akyildiz, I.F., Akan, O.B., Fang, J., "TCP-Planet: A reliable transport protocol for InterPlaNetary Internet", JSAC, vol. 22, no 2, pp. 348361, 2004

[21] Lin, S., Costello, D. J., Miller, M. J.,"Automatic-repeat-request error control schemes", IEEE Commun. Mag., vol. 22, n. 12, pp. 5-17, Dec. 1984

[22] Lun, D. S., Ratnakar, N., Médard, M., Koetter, R., Karger, D. R., Ho, T., Ahmed, E., Zhao, F., "Minimum-Cost Multicast Over Coded Packet Networks", IEEE Trans. on Info. Theory, vol. 52, no. 6, pp.2608-2623, Jun.2006

[23] Chapeau-Blondeau, F., and Monir, A., "Numerical Evaluation of the Lambert W Function and Application to Generation of Generalized Gaussian Noise With Exponent 1/2", IEEE Trans. on Signal Proc., Vol. 50, No. 9, Sept. 2002