Exploiting Sparse Coding: A Sliding Window Enhancement of a Random Linear Network Coding Scheme

Pablo Garrido, David Gómez, Jorge Lanza and Ramón Agüero, *IEEE Senior Member*
Universidad de Cantabria, Santander, Spain
{pgarrido, dgomez, jlanza, ramon}@tlmat.unican.es

Abstract—**Random Linear Network Coding (RLNC) is a technique that provides several benefits. For instance, when applied over wireless mesh networks, it can be exploited to ease routing solutions as well as to increase the robustness against packet losses. Nevertheless, the complexity of the decoding process and the required overhead might jeopardize its performance. There is a trade-off when deciding the field and block sizes; larger values decrease the probability of transmitting linearly dependent packets, but they also increase both the required overhead and the decoding complexity. In order to overcome these limitations, we propose a sliding window enhancement; a fixed number of packets (fewer than the block size) is combined within every transmission, and the decoding process can therefore take advantage of the algebra with sparse matrices. The paper presents an analytical model, which is first validated and later broaden by means of an extensive simulation campaign carried out over the ns-3 simulator.**

Index Terms—**Random Linear Coding; Sparse Matrices; Simulation; Wireless Networks**

I. Introduction

Wireless technologies have undergone a continuous growth in several aspects: number of users, devices, traffic load, requirements. To appropriately tackle this changing environment, they are in constant evolution, and new mechanisms are developed to increase the performance of traditional protocols (for instance, TCP) over these networks.

Network Coding, originally proposed by Ahlswede *et al.* in [1], questions the *store and forward* paradigm and sets out a new network understanding. Packets are not longer immutable entities and data can be combined, re-combined or discarded by nodes through the network. This concept offers different advantages over wireless mesh networks, either with unicast or multicast communications, as well as for data storage services. Some initial works by Koetter and Li [2], [3] showed that the use of linear codes can bring the multicast maximum capacity, while Ho *et al.* [4] proved that random generation of linear codes achieves the optimum performance with high probability, proposing the Random Linear Network Coding (RLNC) scheme.

Afterwards, various studies have assessed the benefits of this solution. In any case, its computational complexity and required overhead is usually overlooked, although it can jeopardize its overall performance. In this sense, there are fewer works proposing solutions to reduce this computational complexity. In this paper we propose a window-based scheme where, instead of coding random packets from the whole corresponding block, we combine a smaller and fixed amount of packets, following a sliding window procedure, at each transmission. We reduce the overhead caused by the transmission of the coding vector, by including a smaller number of coefficients. Besides, the decoding complexity can be reduced, exploiting the algebra with sparse matrices, as was shown by Feize *et al.* in [5].

The novel contributions of this paper are: (1) the proposal of an analytical model of the sliding-window based RLNC, which is validated by means of an extensive simulation study; (2) a thorough analysis of how the operational parameters configuration affect the system performance; (3) a simulation-based study of the performance of the proposed scheme over faulty wireless links; (4) an analysis of the gains brought by recoding at intermediate nodes.

This paper is structured as follows: Section II summarizes the most relevant studies that have tackled this problem. In Section III we briefly describe the RLNC protocol and the proposed window-based scheme; we also introduce an analytical model that can be used to derive its performance. Afterwards, Section IV validates this model, which is also broaden by means of a thorough simulation campaign. Finally, Section V concludes the paper, proving an outlook of the aspects that will be tackled in our future research.

II. Related Work

As was already mentioned, Network Coding (NC) questions the legacy *store-and-forward* paradigm. Previous research has mainly focused on two paths to exploit NC techniques. The first one, referred to as Inter-flow NC, combines packets from different flows, and one of the most well known proposals within this group is the COPE protocol [6]. However, Inter-flow NC has been shown to suffer from a number of issues [7].

The second approach, Intra-flow NC, combines packets belonging to the same data flow. Chachulski *et al.* introduced RLNC in [8] to reduce signaling packets, outperforming previous deterministic coding solutions. There are also some works that have combined the two main realms; for example, Krigslund *et al.* showed in [9] that the RLNC can effectively address some of the the main problems exhibited by COPE; however there is still a remarkable overhead associated to the transmission of coding vectors.

In our previous work we have carried out a thorough evaluation of the RLNC scheme. The architecture of the modules that were implemented in the framework of the `ns-3` was presented in [10], together with a detailed analysis of some of its operational parameters. Afterwards, a probabilistic transmission scheme was introduced in [11], which also studied the benefits of introducing recoding at intermediate nodes.

One of the strongest arguments against the use of RLNC is the one that questions its decoding complexity, which can be roughly estimated as $\mathcal{O}(K^3)$, being K the number of packets to be transmitted per block. This is particularly high, if we compare it to other sparse end-to-end coding schemes, such as LT [12] and Raptor Codes [13], with a decoding complexity of $\mathcal{O}(K \cdot \ln \frac{1}{\epsilon})$. However, these solutions have some limitations, particularly if they are going to be used over wireless mesh networks, since they do not consider the recoding at intermediate nodes, which is one of the most relevant features of the RLNC scheme. Feizi *et al.* [5] proposed a Tunable Sparse Network Coding scheme in order to reduce the complexity of the decoding process, with a density that was dynamically adapted. In short, packets are generated with a density d, which is increased after the transmission of a number of coded packets. This approach was also exploited by Sorensen *et al.* in [14], where they introduced a more advanced scheme.

On the other hand, it is worth highlighting that decoding complexity is not the only problem of RLNC. The overhead due to the coding vector sent in each coded packet shall be also considered. Heide *et al.* [15] analyzed the impact of different parameters (block, field size and density) over the RLNC overhead, with an approach similar to the one used in [5], since it combines a small number of packets (density) in each transmission. However, their study overlooked the performance of their proposed scheme and did not consider lossy channels nor the potential gains that might be brought by the use of recoding.

As mentioned earlier, in this work we address these limitations, and the performance of the proposed solution is thoroughly analyzed by means of a simulation campaign over the `ns-3` simulator [16].

III. Design and implementation

A. RLNC scheme

The RLNC module is implemented as a new layer, placed between IP and UDP. At the source node, it stores packets received from the upper layers (UDP) in a *transmission buffer* until it has received K packets; said in other words, the (fixed) block size equals to this K parameter. Then, the source starts sending coded packets as random linear combinations of such packets, which belong to the same block, $p'_j = \sum_{i=0}^{K-1} c_i \times p_i$. These coefficients are randomly selected from a Galois Field, $GF(2^q)$ and we can therefore associate a coding vector, $\vec{c_j}$, to each coded packet, entailing all the coefficients used during the coding process. Both the Galois Field and block sizes are parameters with a direct impact over the performance. The source keeps transmitting these combinations until it receives an acknowledgment from the destination, when it moves to the next block.

The destination incorporates two storage entities: a *reception buffer*, which can keep up to K packets, and a *decoding matrix*, $\mathcal{C}(K \times K)$, which is built with the received coding vectors. When the destination receives a packet, the coding vector is obtained from the header and is stored in the *decoding matrix*. If its rank increases, i.e. the vector is linearly independent from the ones previously received, the packet is kept at the *reception buffer*, and otherwise it is silently discarded. Once the destination has received K linearly independent packets it decodes the whole block, sending an acknowledgment to the source node.

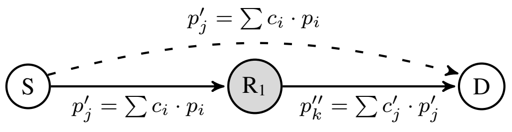

$p'_j = \sum c_i \cdot p_i$
S
R$_1$
D
$p'_j = \sum c_i \cdot p_i$
$p''_k = \sum c'_j \cdot p'_j$

Figure 1: Operation of the *RLNC* scheme over a three-node chain topology

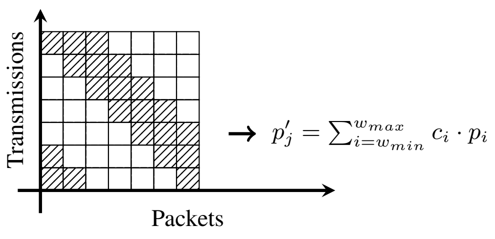
The operation of the intermediate nodes is similar to that of the source; the RLNC scheme provides some "intelligence" and packets are processed and could be as well discarded. First, the coding vector is obtained to assess whether it is linearly dependent from the ones previously received. In such case, the packet is discarded, and otherwise it is stored in the *transmission buffer*. Then, the intermediate nodes send recoded packets, by randomly combining the previously stored packets.

In Figure 1 a canonical topology is shown as an illustrative example. Using the traditional *store and forward* scheme all the packets that the destination overhears over the direct link with the source are discarded, and only those that have been forwarded by the intermediate node are useful; those have an overall loss probability of $1 - [(1 - FER_{S,R_1}) \cdot (1 - FER_{R_1,D})]$, where $FER_{i,j}$ is the Frame Error Rate of the link i, j. On the other hand, the recoding feature can limit the impact of transmission losses; if we assume that the intermediate node can generate packets even if it has not received any from the source, the overall loss probability is established by the worst channel, $\max(FER_1, FER_2)$. Moreover, if the promiscuous mode is enabled at the destination node, it might overhear packets, even if they are not addressed to him, and the information overheard over the direct link $S \rightarrow D$ could be different from that received over the $S \rightarrow R_1 \rightarrow D$ path. Hence, the overall number of transmissions is reduced.

A new header is included in each packet at the NC level, as is depicted in [10]. Note that both the Galois Field and the block sizes are included within this header, as well as the complete coding vector, $\vec{c}$. We have limited the block size to 256, since the latency for larger values would be too long; furthermore, the field size is bound to 8, so as to respect the limitations imposed by the M4RIE library [17], which is used in our implementation.

B. Sliding Window Based Coding

Various works ([5], [14]) have shown that the use of sparse coding vectors might reduce the computational complexity of the decoding process at the destination. In this study, we also pay attention to both reducing the overhead and increasing the throughput. We introduce a new parameter in the RLNC scheme, the sliding window size w, and rather than sending random linear combinations of the whole block, the source node transmits combinations of packets that belong to the same sliding window, w. As can be seen in Figure 2, it starts with $w_{min} = 0$ and $w_{max} = w$ and both indexes are increased with one packet after every transmission.



Transmissions
$\rightarrow p'_j = \sum_{i=w_{min}}^{w_{max}} c_i \cdot p_i$
Packets

Figure 2: Illustrative example of the coding process for a 3-packet window; each row corresponds to a single transmission, entailing 3 different packets; the columns reflect the packets in which the block has been divided (7 in this case)

In order to find the exact probability of a successful block reception (decoding process) of K packets after having received N, $\mathbb{P}_{NK}$, we use an urn-based model, as was done in [18], [19]. Consider an urn with all the vectors with length K that can be generated in a Galois Field of size $Q = 2^q$. If we discard the null coefficients, there would be $(2^q - 1)^K$ possible vectors. Under this scheme, we can guarantee that the first $K - (w - 1)$ generated vectors are linearly independent, since they always include a novel packet. In order to decode the whole block, the destination needs to receive K linearly independent coding vectors. Since we know that the first $K - (w - 1)$ ones are independent, the problem can be reformulated, as the generation of $w - 1$ linearly independent packets in the next $N - K - (w - 1)$ tries. We can therefore see the problem as a linear system where we have to generate w "equations" for w unknown "variables". Since the already $K - (w - 1)$ generated equation are linearly independent, we can build an initial "equation" as a linear combination of the existing ones.

We will first consider the case where $N = K$, and then extend it for N greater than K.

Case N = K: As was already said, after taking $K - (w - 1)$ coding vector from the urn, we can artificially build a "new" coding vector of w coefficients. Then, the probability of decoding the whole block after receiving K packets can be seen as the probability of extracting $w - 1$ new coding vectors from such urn. Since there are $2^q - 1$ linearly dependent vectors and we can overall generate $(2^q - 1)^w$ different vectors, the probability of obtaining a new linearly independent vector equals, for the first try, $1 - \frac{2^q-1}{(2^q-1)^w}$.

$$\begin{bmatrix} \alpha_{K-2} & \alpha_{K-1} & \alpha_0 \\ \beta_{K-2} & \beta_{K-1} & \beta_0 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} \alpha'_{K-1} & \alpha'_0 & \alpha'_1 \\ \beta_{K-1} & \beta_0 & 0 \\ \gamma_{K-1} & \gamma_0 & \gamma_1 \end{bmatrix}$$

Figure 3: Illustrative example of a coding matrix update

In order to illustrate the process, Figure 3 shows an example where the window size is 3. The first extraction is represented by the vector $\alpha = [\alpha_{K-2} \;\; \alpha_{K-1} \;\; \alpha_0]$, built as a linear combination of the $K - (w - 1)$ previously received vectors, keeping, as unknown variables, the last two packets of the block as well as the first one: $K - 2$, $K - 1$, 0, since the next coded packet would involve those indexes. As mentioned earlier, the probability for the next extraction to be linearly independent is $1 - \frac{2^q-1}{(2^q-1)^w}$. Afterwards, the relevant coefficients would affect the $k - 1$, 0 and 1 packets, and the problem can be therefore rewritten as can be seen in Figure 3, where the new vector α' is built from the combination of the $K - (w - 1)$ initial vectors, such that the unknown variables are those corresponding to the $K - 1$, 0, 1 packets.

Hence, there will be $(2^q - 1)^2 - (2^q - 1)$ linearly dependent vectors from α' and β', and the probability of obtaining a third linearly independent vector would be $\frac{(2^q-1)^2-(2^q-1)}{(2^q-1)^w}$; finally, the probability of having K linearly independent vectors after $N = K$ extractions (receptions) is given by:

$$\mathbb{P}(N, K) = 1 + \frac{(2^q - 1)}{(2^q - 1)^w} \prod_{j=2}^{w-1} \frac{(2^q - 1)^j - (2^q - 1)^{j-1}}{(2^q - 1)^w} \quad (1)$$

Case N > K: Lets first assume $N = K + 1$ extractions. We can again assume that the first vector is always independent (probability $\mathbb{P}_1 = 1$), since it is obtained by linearly combining the $K - w$ initial receptions. In the second extraction, the probability of obtaining a linear dependent vector will be $\mathbb{P}_2 = \frac{2^q-1}{(2^q-1)^w}$ and in such case, there are still $w-1$ extractions to obtain $w - 1$ independent vectors. Note that if the k^{th} reception is already linearly dependent, all the remaining ones must be independent ($N = K + 1$). For the case of a 3-packet window, the final result is that shown in Eq. 2, where $\mathbb{P}_2$ and $\mathbb{P}_3$ are the probabilities of obtaining a linear dependent vector in the second and third receptions, respectively.

$$\mathbb{P}(K + 1, K) = \mathbb{P}(K, K) \cdot (1 + \mathbb{P}_2 + \mathbb{P}_3) \quad (2)$$

If we follow the same reasoning for a generic N, we can finally obtain the following result.

$$\mathbb{P}(N, K) = \mathbb{P}(K, K) \left[1 + \sum_{r_1=2}^{w} \mathbb{P}_{r_1} \left[1 + \sum_{r_2=r_1}^{w} \mathbb{P}_{r_2} \right.\right. \\ \left.\left. \ldots \left[1 + \sum_{r_{N-K}=r_{N-K-1}}^{w} \mathbb{P}_{r_{N-K}} \right] \right] \right] \quad (3)$$

The density of the sparse code is defined as the ratio of non-zero coefficients in the coding matrix. Hence, in our case, $d = \frac{w}{K}$. There exits a trade-off between the density (lower densities lead to less complex decoding operations), and the overhead associated to the number of linearly dependent packets that are transmitted.

Last, it is worth mentioning that thanks to the sliding-window approach, we reduce the overhead of the RLNC header. As has been already seen, the header size depends on the Galois field and block sizes, $\lceil \frac{K \cdot q}{8} \rceil$. It is known that larger Galois Field sizes are able to reduce the transmission of linearly dependent packets. However, the overhead required to include the coefficients in the corresponding header might be unacceptable, especially for larger block sizes. This is avoided in the proposed window-based scheme, since the corresponding header carries fewer coefficients (corresponding to the packets that are combined in a transmission).

IV. Results

This section first assesses the performance of the proposed scheme using the theoretical model that was previously discussed, validating its correctness. Afterwards, the analysis

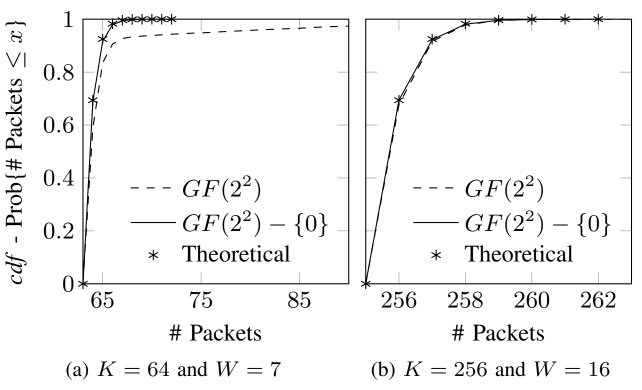


Figure 4: cdf of number of transmitted packets to ensure a successful block decoding

is broaden, by means of an extensive simulation campaign carried out over the `ns-3` simulator.

The common parameters for every simulation are a transmission of 3200 packets and the link layer is set up following the IEEE 802.11b (11 Mbps) standard. We study the performance over two scenarios: first, we use a single wireless link; and afterwards we add a third node to build a chain topology, mimicking the one previously seen in Figure 1 so as to analyze the impact when the intermediate node recodes packets before forwarding them. In our previous papers [10], [11] we have already studied the impact of the field and block sizes, and based on the outcome of such works, we limit the block size to two values: $K = 64, 256$, since they led to the best performances; in addition, we disable MAC-layer retransmissions, since we also proved that they do not provide any benefit, when working with the RLNC scheme.

A. Overhead Analysis

In a first set of experiments, we verify the validity of Equations 1 and 3. After a large number of runs (1000) Figure 4 shows the cumulative distribution function (cdf) of the number of packets that need to be transmitted by the source, under ideal conditions (no packets are lost within the wireless channel), for a successful decoding process. The field size is $q = 2$ and we represent the results for two distinct cases: when the coefficients are randomly chosen from the whole $GF(2^q)$ (*Zeros*) or when we discard *null* values (i.e. $GF(2^q) - \{0\}$, *No Zeros*). The figure also includes the values that are obtained by plotting the results of Eq. 3, showing a perfect match for the *No Zeros* case. The results show a worse behavior for smaller field and window sizes, as can be seen in Figure 4a, which also reflects a strong impact of *null* coefficients. For higher values of these parameters, there is not a relevant difference between the two schemes.

In order to evaluate the impact of the particular configuration of block, field and window sizes in terms of the overhead, we introduce two additional parameters. The percentage of packets that are discarded due to linear dependencies is defined by $\epsilon = \frac{N-K}{N}$, and it has a clear impact over the performance. Besides, it is also important to take into account the overhead induced by the RLNC header, H_{RLNC}. In this sense, there might be certain configurations in which the number of linear dependencies are reduced, but this does not compensate the additional header length, and the overall overhead is, in fact, increased. Thus, we define the overall overhead, ξ, in Eq. 4, as a function of the total number of spurious packets ($N - K$) and the corresponding header and packet lengths, H_{RLNC} and $\mathcal{L}$, respectively.

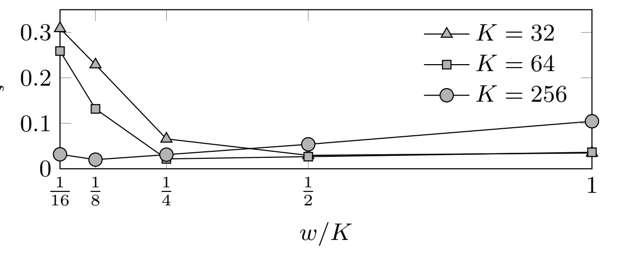


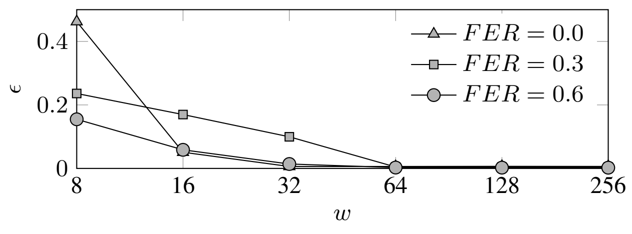
Figure 5: Total overhead

$$\xi = \frac{N \cdot H_{RLNC} + (N - K) \cdot \mathcal{L}}{K \cdot \mathcal{L}} \tag{4}$$

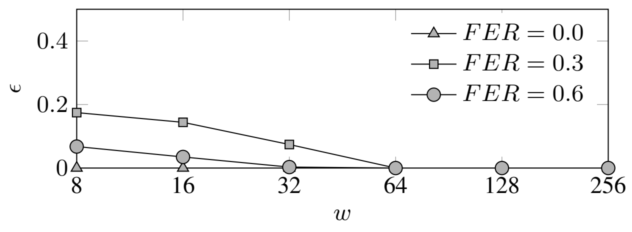
Figure 5 shows the evolution of the overall overhead (ξ) as a function of the window size and for different block lengths. In order to fairly compare the different block sizes, the x-axis corresponds to the ratio between the window and the block size; as can be seen the performance depends on this ratio, rather than on the window value itself. The use of small window values yields a remarkable increase of the number of spurious transmissions, since the probability of receiving linearly dependent packets becomes quite high, as can be seen from the analytical model. For $K = 256$ we can see an interesting effect, since there is an optimum value for the window size, and larger windows would actually lead to higher overheads. This reflects a longer RLNC header without a relevant decrease of the spurious packets. By inspecting the obtained results, we can establish the most favorable configuration, which would be $K = 256$ and $w = 32$, yielding an overhead of $\xi \approx 0.02$, for any field size.

In order to analyze the behavior when $K = 256$ over a non ideal scenario, we randomly drop packets while they are being transmitted, configuring a Frame Error Rate (FER) that is varied between 0.0 and 0.6. In this case, we cannot use the overhead to assess the performance of the scheme, since the increase of transmissions is not only a consequence of linear dependencies, but packets can be also lost during their transmission. In this sense, we study the evolution of the ϵ ratio, which is shown in Figure 6. Besides, we assess the performance for different field sizes[1]. We can again see that the linear dependencies decrease until a certain window size is used $w = K/8 = 32$ for the error-free scenario. In this case, it is worth highlighting the impact of packet losses; when the window is long enough, $w > K/4$, the linear dependencies are not affected by packet losses. However, for shorter window sizes, the sliding window scheme has an interesting effect, since having more dropped packets can even reduce the number of linearly dependent received packets. For smaller *FER* values, if either of the two last packets are lost, the source needs to go over the complete block (quite likely transmitting useless packets); on the other hand, when the

[1]Note that the binary configuration $q = 1$ is the only case in which *null* coefficients are allowed.



(a) $K = 256$, $GF(2)$



(b) $K = 256$, $GF(2^8)$

Figure 6: Ratio between linear dependencies and total number of received packets

FER is higher, many packets might have been previously lost, and therefore not so many redundant receptions are observed.

B. Performance Analysis

First, we obtain an analytical result, establishing the throughput perceived by the RLNC layer, which corresponds to that offered by the UDP protocol. We calculate it by means of the well known Bianchi's model [20]. In addition, this baseline performance is jeopardized by a number of different factors that can be estimated.

The first one corresponds to the transmission of linearly dependent coding vectors, $\vec{c}$, which are silently discarded and thus have a negative impact over the system performance. Eq. 3 can be used to derive the probability of a successful decoding process after having received N encoded packets, so the average number of required transmissions can be easily obtained, as shown in Eq. 5, where $p_{dc}(K,N)$ is the corresponding probability density function, which can be computed as $p_{dc}(K,N) = \mathbb{P}(K,N) - \mathbb{P}(K,N-1)$.

$$E[N] = \sum_{i=K}^{\infty} i \cdot p_{dc}(K,i) \tag{5}$$

As it is clear that $E[N] \geq K$, we can finally establish the average excess packet ratio, ϵ, which will have a negative impact over the system throughput.

$$\epsilon = \frac{E[N] - K}{E[N]} \tag{6}$$

Another aspect that might jeopardize the performance is the transmission of the acknowledgment packet by the receiver when it successfully decodes a block. If we define τ as the average delay of a data packet, and τ_{ACK} as the one corresponding to an acknowledgment, the penalization factor caused by the these confirmation packets, ϵ_{ACK}, can be estimated as follows.

$$\epsilon_{\text{ACK}} = \frac{\tau_{\text{ACK}}}{E[N] \cdot \tau + \tau_{\text{ACK}}} \tag{7}$$

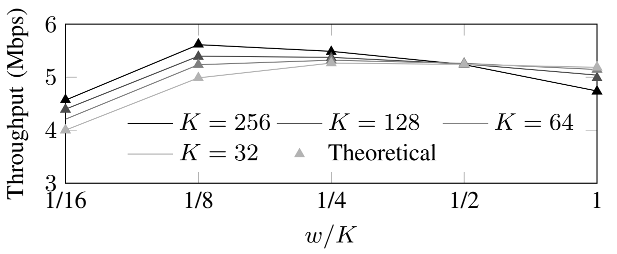


Figure 7: Throughput over a single link communication, $FER = 0.1$

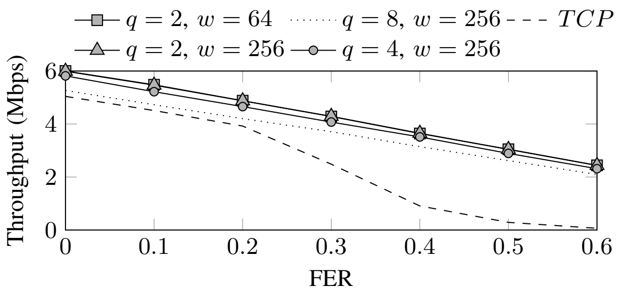


Figure 8: Throughput Vs. Link quality (*FER*)

Finally, the overall throughput, $S = S_{\text{max}} \cdot (1-\epsilon) \cdot (1-\epsilon_{\text{ACK}})$, where S_{max} is the throughput under saturation conditions (Bianchi's model), over error-free links.

Figure 7 shows the throughput over a single link between two nodes with $FER = 0.1$, for different block and window sizes. As could have been expected, after the discussion of the previous section, the best performance is achieved for the largest block size, and when the window size equals 32. In addition, we can observe that, for all cases the sliding window RLNC outperforms the original RLNC scheme (it corresponds to the value $w/K = 1$). We can also highlight the good match between the theoretical values and those obtained during the simulation campaign.

We now assess the performance over a lossy channel, fixing its quality by increasing the FER between 0.0 and 0.6. We focus on a block size of $K = 256$, since we previously showed that the best performance was achieved with this configuration. The results yield that a window size of 64 leads to the highest throughput, no matter the field size, as shown in Figure 8. We obtain the same performance than that best one of the original scheme (i.e. without sliding window, $w = 256$) [10]. In addition, we shall also mention that the proposed RLNC scheme always outperforms the legacy TCP protocol, especially when the link quality get worse.

Finally, in order to evaluate the impact of the recoding feature, we use the chain topology (see Figure 1); we fix a $FER = 0.6$ for the direct link between S and D, while the two other links, $S \rightarrow R_1$ and $R_1 \rightarrow D$, have the same quality ($FER = 0.1$). Under these circumstances, a legacy routing scheme would likely select the $S \rightarrow R_1 \rightarrow D$ path. In addition, if the relaying node R_1 did not recode the packets before forwarding them, the packets overheard by node D over the direct link with the source node would very likely be redundant, and the destination would not be therefore able to take advantage from such receptions. On the other hand, recoding at R_1 would actually increase the probability that

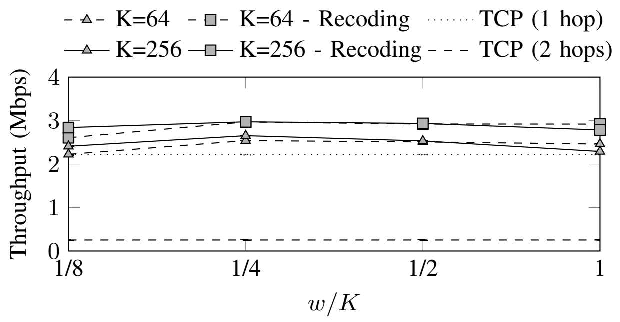


Figure 9: Throughput over the 3-node chain topology ($S \rightarrow R_1 \rightarrow D$) for different configurations

packets sent by R_1 bear different information[2].

In Figure 9 we can see that the gain brought by enabling the recoding process at the intermediate nodes is $\approx 17\%$. Although the recoding procedure would increase the density as well as the corresponding overhead (more coefficients within the coding vector), its combination with the sparse coding yields this additional performance enhancement, that is almost independent of the window size. The figure also shows the throughput observed for the legacy TCP protocol, using both the direct link ($S \rightarrow D$) as well as the two-hop connection through R_1; in both cases, the performance is lower than the one exhibited by RLNC.

V. Conclusions and future work

In this paper we have discussed the design and implementation of an enhanced coding scheme, based on a sliding window solution. We have derived an analytical model to establish the overhead of the proposed solution, as a function of the block, field and window sizes. The model was first validated and then complemented by means of a thorough simulation campaign carried out over the `ns-3` simulator.

The results showed that there is a trade-off between the three studied parameters, i.e. $GF(2^q)$, K and w. We can conclude that the use of large block sizes and low finite fields would provide the best performance; in addition, we have also seen that the window size can be reduced without jeopardizing the performance, until $w = K/4$. The sliding window based coding scheme yields a lower overhead and reduces the decoding complexity, since the decoding matrix *sparsity* would allow using optimized inversion algorithms, as the one presented in [5]. In addition, we have also seen that the proposed solution can be combined with the recoding of packets at the intermediate nodes, which can yield an additional gain.

In our future work we plan to broaden this analysis. For instance, we did not study the observed latency; there might be applications or services (e.g. real-time applications, video streaming) imposing additional requirements, which could eventually limit the block size (a higher block size would certainly increase the latency). Hence, it would be interesting to derive an analytical model able to establish the optimum configuration to address the different requirements (latency, performance). We would also like to introduce some of the concepts and ideas presented in [14], where the authors proposed extracting and inserting the packets inside a block as they are received.

[2]For a more detailed discussion on the benefits of the recoding procedure, the reader might refer to [8], [11]

Acknowledgements

This work has been supported by the Spanish Government by its funding through the project **COSAIF**, *"Connectivity as a Service: Access for the Internet of the Future"* (TEC2012-38754-C02-01).

References

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, Jul 2000.
[2] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003. [Online]. Available: http://dx.doi.org/10.1109/TNET.2003.818197
[3] S.-Y. Li, R. Yeung, and N. Cai, "Linear network coding," *Information Theory, IEEE Transactions on*, vol. 49, no. 2, pp. 371–381, Feb 2003.
[4] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," *IEEE International Symposium on Information Theory, 2003. Proceedings.*, p. 7803, 2003.
[5] S. Feizi, D. E. Lucani, and M. Médard, "Tunable sparse network coding," in *Proc. of the Int. Zurich Seminar on Comm*, 2012, pp. 107–110.
[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 497–510, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1109/TNET.2008.923722
[7] D. Gomez, S. Hassayoun, A. Herren, R. Aguero, and D. Ros, "Impact of network coding on TCP performance in wireless mesh networks," in *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, Sept 2012, pp. 777–782.
[8] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 169–180, Aug. 2007. [Online]. Available: http://doi.acm.org/10.1145/1282427.1282400
[9] J. Krigslund, J. Hansen, M. Hundeboll, D. Lucani, and F. Fitzek, "CORE: COPE with MORE in wireless meshed networks," in *Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th*, June 2013, pp. 1–6.
[10] D. Gómez, E. Rodríguez, R. Agüero, and L. Muñoz, "Reliable communications over wireless mesh networks with inter and intra-flow network coding," in *Proceedings of the 2014 Workshop on Ns-3*, ser. WNS3 '14. New York, NY, USA: ACM, 2014, pp. 4:1–4:8. [Online]. Available: http://doi.acm.org/10.1145/2630777.2630781
[11] D. Gomez, P. Garrido, E. Rodriguez, R. Aguero, and L. Munoz, "Enhanced opportunistic random linear source/network coding with cross-layer techniques over wireless mesh networks," in *Wireless Days (WD), 2014 IFIP*, Nov 2014, pp. 1–4.
[12] M. Luby, "LT codes," in *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, 2002, pp. 271–280.
[13] A. Shokrollahi, "Raptor codes," *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2551–2567, June 2006.
[14] C. Sorensen, D. Lucani, F. Fitzek, and M. Medard, "On-the-Fly Overlapping of Sparse Generations: A Tunable Sparse Network Coding Perspective," in *Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th*, Sept 2014, pp. 1–5.
[15] J. Heide, M. Pedersen, F. Fitzek, and M. Medard, "On code parameters and coding vector representation for practical RLNC," in *Communications (ICC), 2011 IEEE International Conference on*, June 2011, pp. 1–5.
[16] "The ns-3 network simulator," http://www.nsnam.org/.
[17] M. R. Albrecht, "The M4RIE library for dense linear algebra over small fields with even characteristic," *CoRR*, vol. abs/1111.6900, 2011. [Online]. Available: http://arxiv.org/abs/1111.6900
[18] O. Trullols-Cruces, J. Barcelo-Ordinas, and M. Fiore, "Exact decoding probability under random linear network coding," *Communications Letters, IEEE*, vol. 15, no. 1, pp. 67–69, January 2011.
[19] P. Garrido, D. Gomez, R. Aguero, and L. Munoz, "Performance of random linear coding over multiple error-prone wireless links," *Communications Letters, IEEE*, vol. 19, no. 6, pp. 1033–1036, June 2015.
[20] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 3, pp. 535–547, March 2000.