Performance and Complexity of Tunable Sparse Network Coding with Gradual Growing Tuning Functions over Wireless Networks

Pablo Garrido†, Chres W. Sørensen¶, Daniel E. Lucani¶, Ramón Agüero†

†University of Cantabria
Santander, Spain
{pgarrido,ramon}@tlmat.unican.es

¶ Department of Electronic Systems
Aalborg University, Denmark
{cws,del}@es.aau.dk

***Abstract*—Random Linear Network Coding (RLNC) has been shown to be a technique with several benefits, in particular when applied over wireless mesh networks, since it provides robustness against packet losses. On the other hand, Tunable Sparse Network Coding (TSNC) is a promising concept, which leverages a trade-off between computational complexity and goodput. An optimal density tuning function has not been found yet, due to the lack of a closed-form expression that links density, performance and computational cost. In addition, it would be difficult to implement, due to the feedback delay. In this work we propose two novel tuning functions with a lower computational cost, which do not highly increase the overhead in terms of the transmission of linear dependent packets compared with RLNC and previous proposals. Furthermore, we also broaden previous studies of TSNC techniques, by means of an extensive simulation campaign carried out using the `ns-3` simulator. This brings the possibility of assessing their performance over more realistic scenarios, e.g considering MAC effects and delays. We exploit this implementation to analyze the impact of the feedback sent by the decoder. The results, compared to RLNC, show a reduction of 3.5 times in the number of operations without jeopardizing the network performance, in terms of goodput, even when we consider the delay effect on the feedback sent by the decoder.**

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

I. Introduction

We are witnessing a continuous evolution of wireless technologies and networks. One of the most relevant challenges is the growing traffic demand. In the year 2020, it is expected that the number of Internet connected devices will increase, reaching around 50 billion (there are currently 25 billion devices, according to Cisco estimation [1]). Nevertheless, the highly heterogeneous systems, in terms of computational power, energy and network resources, etc., brings additional challenges to tackle with. Hence, the scientific community is proposing novel techniques to improve the behavior of wireless networks. One example of such solutions is Network Coding (NC).

Network coding, originally proposed by Ahlswede *et al.* in [2], fosters an alternative paradigm to *store and forward* and sets out a new network understanding. Packets can be algebraically combined, re-combined or discarded by nodes while they traverse the network. Some initial works, by Koetter and Li [3], [4], showed that the use of linear codes could yield the optimum multicast capacity, while Ho *et al.* [5] proposed the RLNC scheme, showing that random linear combinations of packets could lead to the optimal performance in terms of network capacity.

Afterwards, different studies have analyzed the benefits of RLNC solutions, showing that NC can bring a more efficient usage of network resources. They also highlighted the possibility of recoding packets at intermediate nodes. On the other hand, their computational complexity and the required overhead are usually overlooked, although they could in fact jeopardize their overall goodput.

In this paper we extend the proposal made by Feizi *et al.* in [6], where they show the potential of enabling a tunable density mechanism, which they coined Tunable Sparse Network Coding (TSNC). TSNC fosters the transmissions of *generations* similar to RLNC, but does not choose a single density for the entire process. When a block starts to be sent, coded packets are generated with a low density, that is to say few original packets are combined. Afterwards, this density is increased throughout the transmission. Using low densities alleviates both the encoding and decoding computational complexity. On the other hand, once the destination node has received a certain number of packets, a low density would nevertheless yield a small probability of receiving innovative packets, so it would be sensible increasing it (i.e. combine more packets) when the number of already received packets gets higher.

The novel contributions of this paper are: (1) the proposal of two novel tuning functions for a practical tunable sparse network coding scheme, which reduces the computational cost of the decoding process; (2) the behavior assessment of the TSNC techniques over a wireless network, exploiting the `ns-3` framework; (3) a study on the feedback sent by the decoder impact on the behavior and on the overall network performance.

This paper is structured as follows: Section II summarizes the most relevant studies related to the research discussed in this paper. Section III briefly describes the RLNC protocol, and the proposed approaches for the TSNC scheme. Afterwards, in Section IV we analyze the complexity of the proposed solution, comparing it a legacy RLNC; this is afterwards broadened by means of a thorough simulation campaign carried out over the `ns-3` framework. Finally, Section V concludes the paper, providing an outlook of the aspects that will be tackled in our future research.

II. Related Work

As was already mentioned, Ho *et al.* introduced RLNC in [5]. They fostered the use of such technique for robust, distributed transmissions and information compression, showing that it was able to outperform previous deterministic coding solutions. Afterwards, a combination of RLNC and an opportunistic routing mechanism was presented by Chachulski *et al.* in [7], leading to the so-called MAC-independent Opportunistic Routing & Encoding (MORE) protocol.

In some of our previous works we have thoroughly studied the RLNC behavior over error-prone wireless networks by means of thorough simulation campaigns over the `ns-3` framework. The corresponding `ns-3` implementation was presented in [8], which also analyzed the impact of some of its operational parameters. In [9] we introduced a probabilistic transmission scheme, also assessing the benefits of recoding at intermediate nodes.

The RLNC decoding complexity can be quite high: $\mathcal{O}(K^3)$, being K the number of packets per generation. It is worth recalling that other sparse end-to-end coding schemes, such as LT [10] and Raptor Codes [11]; when they are designed to recover the original K symbols by any subset of $(1+\epsilon)K$ symbols, their decoding complexity is much lower: $\mathcal{O}(K \cdot \ln \frac{1}{\epsilon},\ \epsilon > 1)$. They rely on carefully designed degree distributions, which are used to decide how many packets are to be coded together. However, due to such coding structure, recoding is rather complicated with these solutions [12]. Hence, they have clear limitations, if they are to be used over wireless mesh networks, since they do not leverage recoding at intermediate nodes, which is one of the most relevant features of the RLNC scheme.

Sparse coding techniques have been exploited in order to reduce such computational complexity. For example, Tassi *et al.* [13] discussed a convex resource allocation framework that allows minimizing the complexity of RLNC. In some cases, decoding complexity is not the only issue of RLNC solutions, since the overhead caused by the coding vector that needs to be embedded in each coded packet shall be also considered. Heide *et al.* [14] analyzed the impact of different operational parameters (generation size, field size and density) over the RLNC overhead; their approach is similar to the one used in [6], combining a small number of packets (low density) in each transmission. Moreover, a more complex scheme where sparse codes and overlapping generations was exploited by by Sørensen *et al.* in [15].

However, a static sparse coding level could lead to a larger number of transmitted coded packets, jeopardizing the network performance. Feizi *et al.* [6] already proposed a TSNC scheme to reduce the complexity of the decoding process, with a density that was dynamically adapted. In short, packets are generated with a certain density d, which is incremented after the transmission of a number of coded packets.

The goal of this work is to optimally establish when and how should the encoder increase the coding density. Feizi *et al* [16] assessed the evolution of the density and its impact over the goodput, using various tuning functions, but considering a synthetic scenario, since they assumed the encoder had full knowledge about the decoder status. Sørensen *et al.* proposed a practical scheme in [17], where the encoder just establishes the sparsity when the decoder has received an amount of packets. In both cases, ideal and reliable (i.e. zero-delay and no erasures) networks are assumed between encoder and decoder, while real wireless channels are considered within this work.

III. Design and Implementation

In this section we start by briefly depicting the operation of both RLNC and TSNC techniques. Afterwards we propose two novel approaches to dynamically adapt the corresponding coding density. Both of them gradually increase the number of coded packets per transmission, so as to find its optimum value, without relying on a perfect feedback from the decoder.

A. System description

The source node stores K packets received from the upper layers in a *transmission buffer*; K corresponds to the *fixed* generation size. Afterwards, it starts sending coded (random linear combinations) packets, belonging to such generation. The corresponding coefficients (c_i) 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, comprising the coefficients used during the coding process. Both the Galois Field and generation sizes are parameters with a direct impact on the overall performance. The source keeps sending combinations until it receives an acknowledgement from the destination, moving to the next generation.

The destination has two storage entities: a *reception buffer*, which can keep up to K packets, and a *decoding matrix*, $\mathcal{C}(K \times K)$, which is populated with the received coding vectors. When the destination receives a packet, the coding coefficients vector is obtained from the header and is stored in the *decoding matrix*. If it is linearly independent from the already received ones, the packet is said to increase the degree of freedoms (*dofs*). Otherwise, it is discarded. Once the destination has received K *innovative* packets, it can restore the original ones, and notifies the sender that it expects a new generation.

The legacy RLNC scheme randomly combines packets from the whole generation and its decoding complexity can be quite high $\mathcal{O}(K^3)$. It has been shown that the use of low density codes might yield an important complexity reduction, as it was exploited by the design of LT [10] or Raptor [11] codes. However, due to the strict coding structure used by these schemes, recoding is rather complicated. TSNC techniques [6] advocate the combination of a small number of packets from the whole generation. We can thus define a set of w packets as $W = \{p_{i_1}, p_{i_2}, ..., p_{i_w} | p_{i_k} \neq p_{i_{k'}}, \forall i_k \neq i_{k'}\}$, randomly selected from the generation. Finally, a $w-$ sparse coded packet is created as the combination of w packets:

$$p'_j = \sum_{i \in W} c_i \times p_i \tag{1}$$

Note that if RLNC was used, then $|W| = K$.

Lets assume i *innovative* packets have been already received, i.e the receiver has i *dofs* out of K. Feizi *et al.* [16] proposed an upper bound for the probability $\mathcal{P}(i, w, K)$ that a new $w-$sparse coded packet is *innovative*:

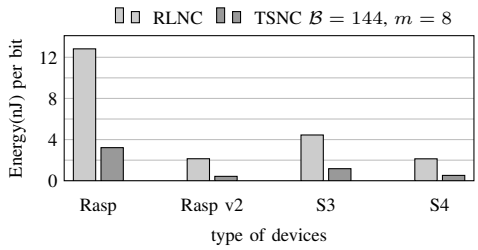


Figure 1: Energy consumption for RLNC and TSNC techniques over different devices

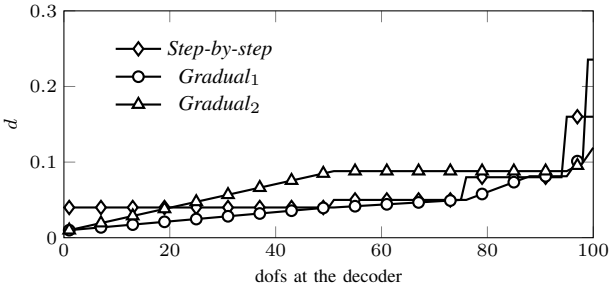


Figure 2: Density evolution of the three approaches

$$\mathcal{P}(i, w, K) \leq 1 - \left(1 - \frac{w}{K}\right)^{K-i} \tag{2}$$

It is worth highlighting that if the decoder has few *dofs*, the probability of receiving an *innovative* packet is high, even for low density combinations. Hence, it is sensible starting with small densities, and gradually increase them as long as the decoder accommodates more linearly independent packets.

We introduce now the *budget*, $\mathcal{B}$, as the average number of packets that are to be transmitted by the source to receive a complete generation at the destination. This budget includes not only the K linearly independent packets, but also the overhead induced by linearly dependent receptions. $\mathcal{B}$ can be calculated based on the previously presented probability as follows:

$$\mathcal{B} = \sum_{i=0}^{K-1} \frac{1}{\mathcal{P}(i, w, K)} \tag{3}$$

In order to stress the relevance of the energy saved when TSNC is used Figure 1 shows the energy consumption of both legacy RLNC and TSNC techniques. These results were presented in [18], where they analyzed the energy consumed by different devices (Raspberry, Raspberry v2, Samsung Galaxy 3, Samsung Galaxy 5), leading to an overall average reduction of around 2.5.

B. Proposed Tuning Functions

Our goal is to establish when the encoder should increase the density, in order to keep the decoding complexity as low as possible, but without increasing the overhead induced by linearly dependent packets. We follow the scheme proposed by Sørensen *et al.* [17]. The decoder provides more frequent feedback when it is close to the end of the generation, where the probability of receiving a linearly dependent packet is higher, and therefore, an appropriate density is crucial. The scheme works as follows: the destination sends a notification back when the corresponding *dofs* equals some predefined values, which depend on the number of notifications established at the decoder, m:

$$s(j) = K \cdot \frac{2^j - 1}{2^j} \qquad j = 0, 1, \cdots, m \tag{4}$$

If we establish a desired budget, $\mathcal{B}$, i.e the total number of coded packets that we pretend to be transmitted by the encoder, we can estimate the density value, $d = \frac{w}{K}$, leading to that specific budget. The value of d can be obtained with the bisection algorithm over the following expression:

$$\mathcal{B}(s(j-1), s(j), K) = \\ = \sum_{i=s(j-1)}^{s(j)-1} \frac{1}{\mathcal{P}(i, w, K)} \leq \sum_{i=s(j-1)}^{s(j)-1} \frac{1}{1 - (1-d)^{K-i}} \tag{5}$$

Given T, the number of already transmitted packets, we can obtain the remaining available budget $\mathcal{B}(s(j-1), s(j), K)$, before shifting to the next level, $\frac{\mathcal{B}-T}{2}$. The encoder can thus select the lowest possible density, provided that it respects the remaining budget. We propose three different approaches to modify the density as the transmission evolves:

- **Step-by-step**: it was presented in [17], and it establishes that the encoder updates the density every time it receives a notification from the decoder, according to Eq. (5) given the desired budget, $\mathcal{B}$, and the number notifications that there decoder would eventually send, m.
- ***Gradual***$_1$: when the encoder receives feedback from the decoder, it calculates the density as was done in the previous alternative, d_1; then, starting from the last known value, the density is linearly increased until it reaches d_1. For example, if the density value that is calculated upon the first notification received from the decoder, d_1, Eq. (5), the density will gradually increase (using a linear trend) from $d = 1/K$ to d_1.
- ***Gradual***$_2$: we introduce the following growth function $d = p \cdot x + b$, where x is the current *dofs*, p corresponds to the slope that would be required to comply with the corresponding budget, and b ensures its continuity. In this case we use the bisection algorithm in Eq. (5) to calculate p, considering the remaining budget, while b ensures that the density does not have discontinuities.

The *Step-by-step* approach has two drawbacks over real networks: (1) notification packets are not instantaneous, and can even be lost; and (2) the large overhead induced by the decoder feedback might jeopardize the overall network performance. In order to have approaches less sensitive to those drawback, the two gradual functions increase the density for every transmitted packet, and do not require a new notification to increase it. Notifications are exploited to establish more appropriate growing functions. In any case, the density never decreases during a generation transmission. If the next value is lower than the previous one, we keep using this latter value.

As an illustrative example, Figure 2 shows the evolution of the three approaches when the budget equals 110, $\mathcal{B} = 110$. The evolution of *Gradual*$_1$ starts from a lower density, $d = 1/k$, and packets are always generated with a lower density

compared with the *Step-by-step* approach. In the second case, *Gradual*$_2$, coded packets are generated with a density similar, on average, to the reference scheme and, although the growing pace is faster at the beginning, the density is actually lower at the end of the transmission.

IV. Results

First we assess the behavior of the TSNC scheme assuming ideal conditions, i.e. the decoder use an error-free and zero-delay connection to send the notifications to the source node. This would allow us to better understand the behavior of the decoding process. Afterwards, we study the decoding complexity as well as the overall network performance, exploiting an implementation of the proposed solutions within the `ns-3` framework. The decoding complexity is obtained through a benchmark implemented using the Kodo library [19], which uses a slightly modified Gaussian Elimination algorithm in the decoding process. The network performance is measured as the goodput at the application layer, i.e the total transmitted bytes divided by the time elapsed from the first transmission by the encoder until the destination gets the complete file.

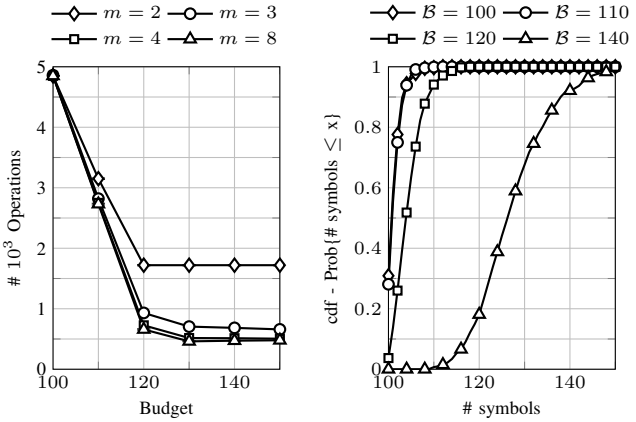
A. Benchmark Results

We are first interested in measuring the number of operations per packet that are required to decode a complete generation; this comparison just focuses on the decoding complexity of the different proposals, without considering the impact of the feedback channel. We have implemented the two enhanced TSNC schemes using the Kodo library [19] and we have carried out a thorough analysis, assuming that both source and the receiver nodes are connected with an ideal (zero-delay) link. We analyze the behavior of three proposals: (1) *Step-by-step*, (2) *Gradual*$_1$ and (3) *Gradual*$_2$

Following the procedure described in Section III, the decoder sends a notification when a particular *dofs* is reached, depending on the number of levels m. We study the corresponding complexity as a function of the number of times the decoder sends those notifications (m) and the allowed budget, $\mathcal{B}$. The encoder is set up with a generation of 100 packets, each of them being 1500 bytes long. All the results are obtained after 1000 iterations, and we show both the average value as well as the 95% confidence interval.

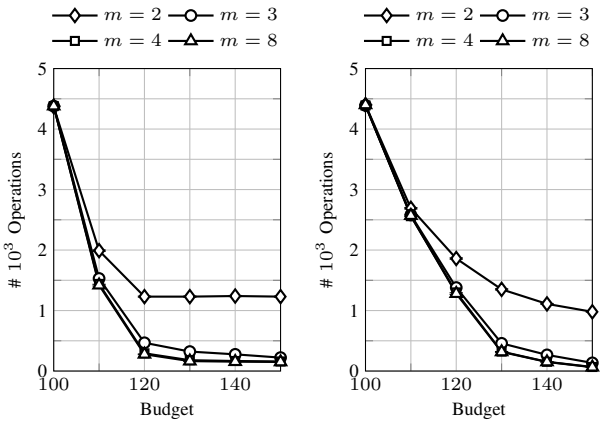
The computational cost of the *Step-by-step* approach can be seen in Figure 3. We study the number of finite field operations required at the decoder and the cumulative distribution function (cdf) of the number of packets that are transmitted by the source node. First, we can see that higher budgets yield lower computational costs in the decoding process, since the encoder can build lower density packets, if more transmissions are allowed. It is also worth highlighting that a small m reduces the possibilities for the encoder to change the density, thus leading to an increase of the number of operations. On the other hand, a high number of levels, $m > 4$, does not reduce the number of operations, since the last density updates happen when the *dofs* at the decoder is close to K.

For the *cdf* of the number of packets that need to be transmitted, we fix $m = 4$, since higher values showed very similar performances. We can see that a small budget, $B = 110$, leads to a relevant reduction of the number of operations (from $\approx 1.53\times$ to $\approx 1.77\times$, depending on the value of m)



(a) *Step-by-step* approach (b) *Step-by-step* approach. $m = 4$

Figure 3: Number of finite field operations and cdf of the number of transmissions Vs. $\mathcal{B}$ and m for the *Step-by-step* function.



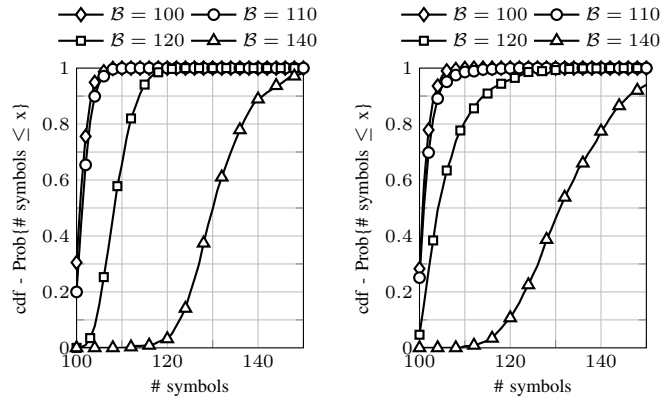
(a) *Gradual*$_1$ approach (b) *Gradual*$_2$ approach

Figure 4: Number of finite field operations Vs. $\mathcal{B}$ and m for the *Gradual* functions

without a significant increase on the number of transmitted packets. However, we can see a clear trade-off between computational cost and network performance, since increasing $\mathcal{B}$ and the overhead caused by linearly dependent packets, severely jeopardizes the overall network performance; we can indeed see that the number of transmissions is much higher.

In Figure 4 we analyze the behavior of the two proposed schemes. As was already explained, *Gradual*$_1$ operation is quite similar to the one fostered by the *Step-by-step* alternative, although it gradually increases the density in every new transmission; on the other hand, *Gradual*$_2$ takes into account the evolution of the density and, instead of estimating the most appropriate value, it establishes the slope of a linear growing function. As shown in Figure 4 the first approach yields a more significant reduction of the number of operations for small budget values, although the second alternative offers a better operation for higher budgets and $m = 2$.

Both approaches have a similar behavior regarding the number of transmitted packets, as can be seen in Figure 5, especially for low budget values. *Gradual*$_1$ generally starts



(a) *Gradual*$_1$ (b) *Gradual*$_2$

Figure 5: cdf of the number of transmissions vs $\mathcal{B}$.

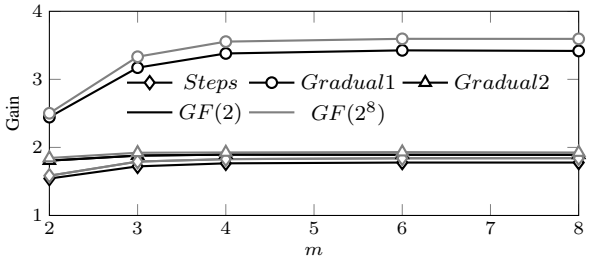


Figure 6: Reduction in the number of operations with respect to the RLNC scheme ($\mathcal{B} = 110$).

from lower densities, $w = 1$, no matter the budget, and gradually increases it until it reaches the value given by Eq. (5), which is in fact an upper bound; as a consequence, the encoder sends lower dense coded packets, without increasing the number of transmissions.

Figure 6 shows the better performance, in the required number of operations, if we take as a reference behavior of a legacy RLNC scheme, i.e $\mathcal{G} = \frac{\#\text{operationsRLNC}}{\#\text{operationsTNSC}}$. We fix the budget, $\mathcal{B} = 110$, since it provides a good trade-off between complexity and overhead, and we also modify the size of the corresponding *Galois Field*, q (for both the TSNC solution and the reference RLNC scheme). We can see that increasing the number of density regions (m) above 4 does not yield any additional benefit; furthermore, the *Gradual*$_1$ approach outperforms the other TSNC schemes. Although the reduce slightly increases for higher q values, we have to bear in mind the larger computational costs, since they require both multiplications and subtractions, compared to the simple `xor` operations if $GF(2)$ is used. Hence, this latter configuration would lead to larger reduction in terms of energy consumption and computational time.

B. Results over realistic wireless links

So far, we have assumed ideal channel conditions, including the instantaneous knowledge of the decoder status at the encoder. However, these assumptions are far from being sensible in a real situation. In order to challenge the proposed schemes under more realistic circumstances, we have implemented the required modules within the framework of the `ns-3` simulator. We have then modeled a single

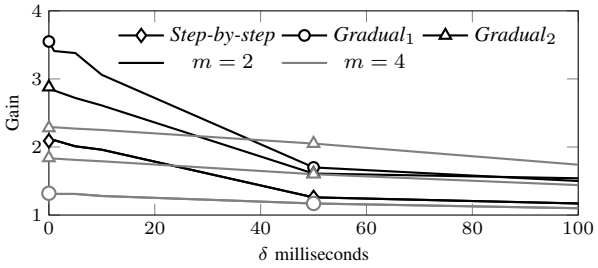


Figure 7: Reduction in the number of operations with respect to the RLNC scheme ($\mathcal{B} = 110$).

wireless link, following 802.11b (11Mbps) recommendation, between the two nodes. Our implementation starts from that presented in [8], where the NC layer is set between the UDP and IP level. Since the decoder needs to receive K arbitrary packets and none of them is particularly relevant, link layer retransmissions are disabled; they do not provide any gain and can be even contra-productive, as was shown in [8]. A maximum number of 4 transmissions per datagram is established when the legacy TCP is used. We have made the changes required to implement the TSNC approach, including the decoder notifications.

First, we study the impact of the delay affecting the *dofs* notifications sent by the decoder. It corresponds to the combination of the propagation delay and the processing time required by the decoder to calculate the *dofs*. In order to capture its impact, we have added a timer within the `ns-3` framework, which is activated whenever the decoder needs to send a notification, to synthetically delay it by δ (*ms*). In this sense, if $\delta = 0$, the delay just corresponds to the propagation time over the wireless link.

We show the reduction in the number of operations that are performed, taking as a reference the ones required by the legacy RLNC, see Figure 7. We keep using a budget of $\mathcal{B} = 110$ packets, and we sweep δ. We can see that the best behavior is obtained by the *Gradual*$_1$ approach, for $m = 4$. Furthermore, if $m = 2$ the impact of δ is less significant, since the encoder has low chances to tune the density, leading to a higher density at the beginning and a similar value after receiving the *dofs* notification from the decoder, no matter when this happens.

Despite the reduction caused by higher δ values, this strongly depends on the particular system and application. The processing time measured in our benchmark is, on average, $0.6\ ms$ per packet. We have carried out the measurements on a computer equipped with an Intel Core i5-3317U, running at 1.70GHz. We can thus state that, in this case, the delay induced by this processing is negligible, since it is much lower than the transmission time.

Once we have seen the complexity reduction offered by the proposed schemes, we next study the network performance, by analyzing the corresponding goodput, as can be seen in Figure 8. We plot the values obtained for the three solutions, comparing them with the performance of the RLNC scheme and the one exhibited by the TCP protocol (using the New Reno version [20]), since it is the traditional transport layer used for reliable services (file transfer). When $\mathcal{B} \leq 110$ the goodput of TSNC mimics the one offered by RLNC and, as

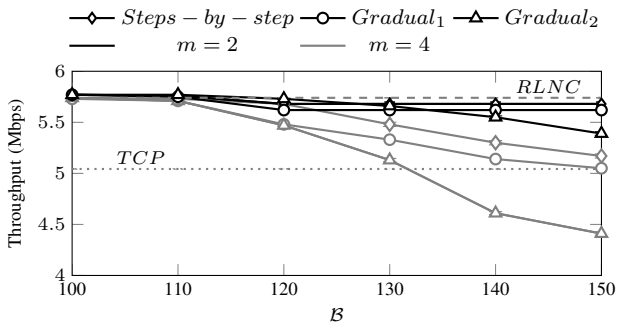


Figure 8: Goodput of the different approaches Vs. $\mathcal{B}$ ($\delta = 0$).

was previously discussed, the computational cost is heavily reduced ($\approx$ 3 times lower). However, when $\mathcal{B} \geq 120$, we can see a goodput reduction, due to the transmission of linear dependent packets.

V. Conclusions and Future Work

In this paper we have proposed two novel tuning functions to be used with TSNC techniques. They aim to reduce the computational complexity of network coding by introducing only a small overhead in the network. Besides, we have broadened previous works, by considering non ideal networks in our analysis. We have implemented the corresponding techniques as well as different tuning functions within the `ns-3` framework and we have exploited such implementation to thoroughly analyze both the goodput and the impact of the corresponding delay in the notifications sent by the decoder.

The results showed a clear trade-off between network performance and computational complexity. However, we can conclude that there is an interesting point where the computational complexity is reduced by $\approx 3.5\times$, without a significant degradation of the goodput. The impact of the feedback delay was also considered in the analysis, and the results showed that it has a clear impact on the computational cost. Note that in any case the TSNC scheme provides better results than the legacy RLNC solution.

In our future work we plan to broaden this analysis. First, we would like to study different tuning functions. The main idea would be to find the most appropriate trade-off between computational cost and network performance. Second, since multicast transmissions is one of the most interesting use cases for the RLNC, we will analyze how the TSNC solution behaves over such networks. In fact, feedback issues get more challenging in multicast networks. Hence, we are interested in proposing a TSNC scheme without a direct feedback from the decoder, as previous works have fostered for RLNC solutions over multicast networks. Moreover, we will also propose a more precise upper bound for the probability of receiving a linearly dependent packet, given the current *dofs*, to improve the density estimation.

Acknowledgements

This work has been supported by the Spanish Government (Minesterio de Economía y Competitividad, Fondo Europeo de Desarrollo Regional, FEDER) by means of the projects **COSAIF**, *"Connectivity as a Service: Access for the Internet of the Future"* (TEC2012-38754-C02-01), and **ADVICE** (TEC2015-71329-C2-1-R). This work was also financed in part by the **TuneSCode** project (No. DFF 1335-00125) granted by the Danish Council for Independent Research.

References

[1] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, pp. 1–11, 2011.

[2] 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.

[3] 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

[4] 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.

[5] 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.

[6] 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.

[7] 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

[8] 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

[9] 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.

[10] M. Luby, "LT codes," in *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, 2002, pp. 271–280.

[11] A. Shokrollahi, "Raptor codes," *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2551–2567, June 2006.

[12] S. Puducheri, J. Kliewer, and T. E. Fuja, "The design and performance of distributed lt codes," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3740–3754, Oct 2007.

[13] A. Tassi, I. Chatzigeorgiou, and D. E. Lucani, "Analysis and optimization of sparse random linear network coding for reliable multicast services," *IEEE Transactions on Communications*, vol. 64, no. 1, pp. 285–299, Jan 2016.

[14] 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.

[15] 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.

[16] S. Feizi, D. E. Lucani, C. W. Sørensen, A. Makhdoumi, and M. Médard, "Tunable sparse network coding for multicast networks," in *Network Coding (NetCod), 2014 International Symposium on*, June 2014, pp. 1–6.

[17] C. W. Sorensen, A. S. Badr, J. A. Cabrera, D. E. Lucani, J. Heide, and F. H. P. Fitzek, "A Practical View on Tunable Sparse Network Coding," in *European Wireless 2015; 21th European Wireless Conference; Proceedings of*, May 2015, pp. 1–6.

[18] C. W. Sørensen, A. Paramanathan, J. Guerrero, M. V. Pedersen, D. E. L. Roetter, and F. Fitzek, "Leaner and meaner: Network coding in simd enabled commercial devices," in *IEEE Wireless Communications and Networking Conference. Proceedings*, 2016.

[19] M. V. Pedersen, J. Heide, and F. H. Fitzek, "Kodo: An open and research oriented network coding library," in *Networking 2011 Workshops*. Springer, 2011, pp. 145–152.

[20] T. Henderson, S. Floyd, A. Gurtov, and Y. Nishida, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 6582 (Proposed Standard), Internet Engineering Task Force, Apr. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6582.txt