Deterministic Broadcasting and Random Linear Network Coding in Mobile Ad Hoc Networks

Nikolaos Papanikos *Student Member, IEEE,* and Evangelos Papapetrou *Member, IEEE*
Department of Computer Science & Engineering, University of Ioannina, Greece
npapanik, epap@cs.uoi.gr

***Abstract*—Network coding has been successfully used in the past for efficient broadcasting in wireless multi-hop networks. Two coding approaches are suitable for mobile networks; Random Linear Network Coding (RLNC) and XOR-based coding. In this work, we focus on the problem of multiple source broadcasting in mobile ad hoc networks. We make the observation that RLNC provides increased resilience to packet losses compared to XOR-based coding. We develop an analytical model that justifies our intuition. However, the model also reveals that combining RLNC with probabilistic forwarding, which is the approach taken in the literature, may significantly impact RLNC's performance. Therefore, we take the novel approach to combine RLNC with a deterministic broadcasting algorithm in order to prune transmissions. More specifically, we propose a Connected Dominating Set (CDS) based algorithm that works in synergy with RLNC on the "packet generation level". Since managing packet generations is a key issue in RLNC, we propose a distributed scheme, which is also suitable for mobile environments and does not compromise the coding efficiency. We show that the proposed algorithm outperforms XOR-based as well as RLNC-based schemes even when global knowledge is used for managing packet generations.**

***Index Terms*—random linear network coding, broadcasting, mobile ad hoc networks**

I. Introduction

BROADCASTING is a cornerstone of many distributed networking protocols in wireless ad hoc networks. From routing [1] to application layer protocols [2], broadcasting is used for distributing and collecting information about the random network. Over the last years, network coding [3] has emerged as an effective approach to enhance the performance of networking protocols. In this context, several researchers have looked into combining network coding and broadcasting in wireless networks [4]–[20]. One line of research focuses on using network coding towards guaranteeing delivery of messages [10]–[20]. This approach is tailored only for static wireless networks mainly due to the cost and the requirements for implementing feedback mechanisms. Moving to a different direction, several efforts [4]–[9] look at the energy and bandwidth consuming nature of broadcasting, which is critical in some types of wireless networks such as ad hoc and mobile ones [21]. Therefore, those schemes take a more energy efficient approach and target at striking the best possible balance between delivery and cost (as expressed by the number of transmissions).

In this work, *we focus on the latter approach which is suitable for ad hoc and/or mobile networks*. Moreover, we are interested in the scenario of *multiple broadcasting sources*, i.e. we examine the many-to-all and all-to-all communication paradigms. Such scenarios frequently appear when multiple nodes independently engage in discovery phases, e.g., in on-demand routing protocols for constructing a path [1], in service discovery applications for finding a resource [2], [22], in peer databases for retrieving volatile data [23], etc.

Based on the coding technique, two approaches can be identified. The first combines *Random Linear Network Coding (RLNC)* [24] with probabilistic forwarding. More specifically, packets are grouped in the so called "generations". Encoded packets are produced as random linear combinations of the packets in a generation, based on the theory of finite fields [25], [26], and then probabilistically forwarded. Receiving enough linear combinations allows the decoding of the original packets. The foundations of this approach have been laid by Fragouli et al. [4]. The key idea is to use RLNC for providing delivery efficiency while probabilistic forwarding alleviates the cost of broadcasting in terms of transmissions. The second approach works on the concept of "coding opportunity" [27] and encodes packets on a hop-by-hop basis using bitwise XOR (*XOR-based coding*). In contrast to the previous approach, the encoded packets are deterministically forwarded and the coding method is oriented towards reducing transmissions rather than coping with transmission failures. The first and most representative algorithm of this category, proposed by Li et al. [7], utilizes XOR-based coding while the encoded packets are forwarded with the partial dominant pruning algorithm [28].

In this work, we first develop an analytical model that captures the performance of coding-based broadcast schemes that focus on energy efficiency. The model confirms that RLNC is a valuable tool for providing resilience to packet failures. However, it also reveals that pruning transmissions, which is an essential process for energy efficiency, may have a significant impact on the effectiveness of RLNC. More specifically, we use the analysis to show that, despite the approach taken in the literature, using probabilistic forwarding to suppress transmissions may significantly impair the performance of RLNC. Therefore, we follow the *innovative approach of integrating RLNC into deterministic broadcasting in order to combine its benefits with a coding-friendly pruning of transmissions*. Below, we summarize our contributions:

- We develop an analytical model (Section IV) that sheds light on the differences between RLNC and XOR based broadcast schemes oriented towards energy efficiency. Such information is very useful since only sparse empirical data exist in the literature for comparing the two methods. The model confirms that RLNC is capable of providing increased resilience to transmission errors.

- We use the developed model to unveil the potential pitfalls of combining RLNC and probabilistic forwarding and stress the need for a topology-aware pruning process.
- Following our observations, we turn to deterministic broadcasting, which has never been used for pruning transmissions in the context of an RLNC enabled scheme. More specifically, the proposed algorithm (Section V) implements CDS (Connected Dominating Set) based forwarding rules "on the generation level" in order to allow the flow of packet generations over the CDS. The rationale is that the CDS will provide a more systematic and topology-aware pruning of redundant transmissions without impairing the coding efficiency of RLNC.
- We address the problem of *generation management*, i.e. the need of nodes to distributively agree in the grouping of packets into generations. Although this is vital for practically implementing RLNC, some of its aspects that are critical when packets from different sources are "mixed" into a generation, are rarely discussed in the literature. We review the pending problems and propose a distributed mechanism that does not compromise the coding efficiency (Section VI).
- Our analytical model also reveals that an increased packet loss rate significantly impairs the performance of RLNC in nodes experiencing poor connectivity. This holds true even if deterministic broadcasting is used. To tackle the problem, we extend the proposed algorithm in order to enhance the topology-awareness of the pruning process (Section VIII).

In the rest of the paper, we discuss the related work in Section II while in Section III we present the design principles of RLNC and XOR-based coding. We evaluate the proposed algorithms through simulation (Section VII and Section VIII) and summarize our findings in Section IX.

II. Related Work

Several studies have investigated the use of network coding for broadcasting in wireless networks. The proposed algorithms can be classified, based on the coding method, into: i) RLNC-based, and ii) XOR-based approaches. RLNC-based algorithms [4]–[6], [10]–[16], [29] build on the concepts of practical RLNC [24]. From this category, only the algorithms that focus on energy efficiency [4]–[6] are suitable for mobile networks. All of them take a probabilistic approach to forward encoded packets. More specifically, Fragouli et al. [4], extend the probabilistic algorithm, proposed in [5], and introduce two topology-aware heuristics to determine the number of encoded packets, that each node should forward, in order for the receivers to decode the original packets. The algorithm also allows the encoding of packets from different sources by incorporating rules for the distributed management of packet generations. Other techniques extend this algorithm by modifying the forwarding heuristics and the generation management mechanism [6]. The second subclass of RLNC-based algorithms [10]–[16] focuses on reliability and integrates some kind of feedback mechanism. The feedback information is used to determine the optimal rate, i.e. the number of packets to be forwarded by intermediate nodes, so that delivery of packets is guaranteed. Clearly, this strategy is not oriented towards minimizing the cost of broadcasting. Furthermore, a feedback mechanism increases the cost while its implementation is not straightforward in mobile networks. Therefore, those algorithms have only been proposed for static networks. Finally, in [29], the authors study the problem of timeliness in broadcasting. They use RLNC over broadcasting trees in a static network and under the assumptions of lossless links and knowledge of global information.

Our approach is novel in that it uses the synergy of RLNC and deterministic broadcasting to improve both resilience to failures and energy efficiency. More specifically, the deterministic algorithm not only forwards packets but also dynamically determines the number of transmissions through a pruning process. This is in contrast with reliability-oriented algorithms that use deterministic broadcasting in the context of a rate selection approach. In those algorithms, the number of transmissions is determined based on feedback information and the deterministic algorithm is just used for forwarding. Furthermore, our approach is distributed and requires only local information and no prior knowledge of the network, therefore it is also suitable for ad hoc and mobile networks.

On the other hand, there are two major subclasses of XOR-based schemes. The first consists of algorithms that adopt the use of rateless codes [18]–[20], such as LT codes [30]. Rateless coding is tailored for guaranteeing message delivery and requires feedback information. Therefore, similar to RLNC-based algorithms with a feedback mechanism, algorithms of this subclass have been proposed only for static networks. The second subclass of XOR-based algorithms [7]–[9], [17], [31] follows the concept of "coding opportunity" [27] to perform coding on a hop-by-hop basis. The prominent algorithm of this subclass, CodeB [7], combines deterministic broadcasting, based on a CDS, with hop-by-hop XOR coding of packets. It also provides information exchange mechanisms that make possible the implementation on mobile environments. The rest of the algorithms in this subclass also employ deterministic broadcasting. However, only a subset of them focuses on energy efficiency in mobile networks [7]–[9], while others focus on reliable broadcasting in static networks [17] or broadcasting with deadlines [31]. Furthermore, they differ in the method used for constructing the CDS [8], [9], [17] and the rules or the buffering scheme used for finding coding opportunities [9], [31]. Hereafter, we use the term "XOR-based coding" to refer to this subclass of algorithms.

III. Preliminaries

RLNC is based on the observation that a linear code, i.e. to linearly combine packets based on the theory of finite fields, is adequate for providing the benefits of network coding [25]. In order to practically implement RLNC, *native*, i.e. non encoded, packets need to be organized in groups, the so called *generations* [24]. Then, an encoded packet is produced as a linear combination of the native packets in a generation, using

$\mathbb{F}_{2^s}$ arithmetic. That is, each native packet p_i is first partitioned into symbols of s bits and then the k-th symbol of the encoded packet $e(k)$ is calculated as $e(k) = \sum_{i=1}^{g} c_i p_i(k), \forall k$, where $p_i(k)$ is the k-th symbol of the i-th native packet and g is the number of packets in a generation. The set of coefficients $\langle c_1, c_2, \ldots, c_g \rangle$, which is called *the encoding vector*, is randomly selected from the finite field $\mathbb{F}_{2^s}$ and appended to the packet header. The random selection provides the required flexibility for distributed implementations. It is also sufficient since the probability of producing linearly dependent packets depends on the field size 2^s [26] and is negligible even for small values of s [32]. Decoding packets of generation i at node v is performed by means of a decoding matrix $\mathbb{G}_{v,i}$. The matrix is populated by *innovative packets*, i.e. the encoded packets that increase the rank of $\mathbb{G}_{v,i}$. Decoding is accomplished by performing the Gaussian elimination when $\mathbb{G}_{v,i}$ has a full rank. It is also possible to decode a subset of packets when a full rank submatrix of $\mathbb{G}_{v,i}$ exists (*partial decoding*). Furthermore, encoding at an intermediate node is possible without the need of decoding the native packets since a new encoded packet may be produced by linearly combining other encoded packets.

In XOR-based coding, each node collects information about the native packets received by its neighbors. The information is collected by overhearing the wireless medium and by exploiting local connectivity information. Let $\mathcal{B}_u$ denote the buffer containing the native packets received by node u and $\mathcal{B}_u^v$ denote v's view of the same buffer. A node u may choose a set of native packets $\mathcal{B}' \subseteq \mathcal{B}_u$ and produce an encoded packet, by using bitwise XOR, in the presence of a *coding opportunity*. This means that a set $\mathcal{B}' \neq \emptyset$ can be found such that, according to u's view, each node $v \in \mathcal{N}(u)$ has received at least $|\mathcal{B}'|-1$ of the native packets in $\mathcal{B}'$, i.e. $|\mathcal{B}_v^u \cap \mathcal{B}'| \geq |\mathcal{B}'|-1, \forall v \in \mathcal{N}(u)$. XOR-based coding works on a hop-by-hop basis, i.e., a receiver of an encoded packet should be able to decode it. Successful decoding depends on the consistency of $\mathcal{B}_v^u$, i.e., whether $\mathcal{B}_v^u{=}\mathcal{B}_v$. Decoding failures occur when $|\mathcal{B}_v \cap \mathcal{B}'|{<}|\mathcal{B}'|{-}1$ and result in the loss of all the encoded packets.

Both RLNC and XOR-based coding entail some communication, processing and storage space overhead. About the communication overhead, both schemes assume that an encoding vector is included in the header of each encoded packet. The processing overhead in RLNC is related to the implementation of the Gaussian elimination. Its complexity on a matrix with rank r is $\mathcal{O}(r^3)$, however, implementing partial decoding can alleviate the decoding cost. On the other had, in XOR-based coding, the processing burden lies in finding coding opportunities. The optimal XOR-based algorithm is shown to be NP-hard, however, efficient suboptimal algorithms for finding coding opportunities have been proposed [7]. Finally, while in RLNC each node is required to store all packets in a generation, in XOR-based coding, each node should store a list of recently received packets (in order to enable decoding) along with information about the packets received by each of its neighbors. To summarize, in our view, none of the above schemes is profoundly better than the other, in terms of the related overheads. Furthermore, the actual cost of each scheme depends on the implementation specifics, making it impossible for a more detailed comparison. Nonetheless, we will show, throughout the rest of the manuscript, that we take all necessary action to minimize the cost of the proposed scheme, e.g. we enable partial decoding, minimize the size of encoding vectors, keep the generation size small, etc.

TABLE I
NOTATION USED IN THE ANALYSIS

g	Generation size
$\mathbb{G}_{v,i}$	Decoding matrix of node v for generation i
N	Number of nodes in network
$\mathcal{N}(v)$	Set of node v's direct neighbors
ω	Probability of forwarding a message
ρ	Probability of transmission failure

IV. Analysis of RLNC's coding features

As mentioned previously, the driving force of this work has been the observation that RLNC is capable of providing robust coding features. To validate this view, we develop an analytical model that portrays the performance of RLNC in the context of broadcasting. Before continuing with the analysis, we briefly describe the system model. Table I summarizes the notation used in the following.

A. System model

Network Model: We consider multihop wireless ad hoc networks. We model such a network as a random geometric graph (RGG) [33]. The nodes are deployed over an area $A \times A$. We focus on the generic approach of uniform node deployment which captures static and some cases of mobile networks (e.g. when node movement follows the random direction model [34]). Moreover, our study is valid for the node distribution resulting from the random waypoint movement model [35]. A link between a node pair (u, v) exists when the Euclidean distance $d(u, v)$ is smaller than a transmission range R. The neighborhood $\mathcal{N}(v)$ of a node v is the set of nodes connected to v with an link, i.e. $\mathcal{N}(v) = \{u \mid d(u, v) \leq R\}$.

Loss Model: The network consists of unreliable links. The transmission of a packet over a link fails with probability ρ, which is independent of other links. This assumption is common in the literature [17], [36] for wireless links without correlated shadowing and severe interference.

Broadcast sources: We assume that multiple sources exist in the network. Created packets are grouped in generations of size g. For each packet added to a generation, the source broadcasts an encoded packet that is a random linear combination of the generation contents.

Forwarding process: When receiving an innovative packet, each node implements a simple probabilistic forwarding protocol, i.e. forwards a new encoded packet with probability ω.

B. Distribution of the number of message copies

The properties of an RGG are critical for the performance of RLNC. More specifically, we will show that the performance of RLNC depends on the number of message copies that a node d receives when a source s broadcasts a message without using network coding. Let us model this number as a discrete random variable (RV), denoted as X. We aim at identifying a good approximation for the probability mass function (pmf) of X. We first assume lossless links (i.e. ρ=0) and later generalize our model to include the case of $\rho\neq 0$. First, note that X is conditional on the number of d's neighbors that receive at least one copy of the broadcast message. If Y is a RV representing the latter number, then X follows the binomial distribution with parameters Y and ω, i.e. $X\sim B(Y,\omega)$. This is because the forwarding decisions made by neighbors are independent. Then, we focus on research efforts that have established, by means of percolation theory, that probabilistic forwarding presents a bimodal behavior [37]. That is, if we consider the number of nodes (r) that receive the message, then, with high probability, either $r=0$ or $r=\alpha, \{\alpha\in\mathbb{N}: 0<\alpha\leq N\}$. The probability that r has any other value is negligible. The actual probability of r=0 (and the complementary of r=α) as well as α depend on the network properties. Moreover, in most cases, $\alpha\to N$, i.e. either none or nearly all the nodes receive the message [37]. By extending this finding, we make the observation that Y also exhibits a near bimodal behavior, therefore a good approximation for Y's pmf is:

$$\mathrm{P}\{Y=k\}=\begin{cases}\phi & k=0\\ 1-\phi & k=|\mathcal{N}(d)|\\ 0 & \text{otherwise}\end{cases} \tag{1}$$

where $\{\phi\in\mathbb{R}: 0\leq\phi\leq 1\}$. The rationale behind this approximation is simple; due to spatial proximity of the nodes that belong to $\mathcal{N}(d)$, all of them will lie either in the set of receivers or in the set of non-receivers with high probability. Using (1), it is easy to show[1] that:

$$\mathrm{P}\{X=k\}=\begin{cases}\phi+(1-\phi)(1-\omega)^{|\mathcal{N}(d)|} & k=0\\ (1-\phi)\omega^k(1-\omega)^{|\mathcal{N}(d)|-k} & 0<k\leq|\mathcal{N}(d)|\end{cases} \tag{2}$$

To validate this distribution, we need to examine it under various combinations of ω, $|\mathcal{N}(d)|$, the average node degree and the hop distance (H) between s and d. This is because ϕ, in analogy to the bimodal property [37], also depends on those parameters. Therefore, we adopt the following strategy; we simulate probabilistic broadcasting for various values of ω in RGGs deployed in areas of various sizes (we use the normalized value $\hat{A}$=A/R to denote the size of the network area). Then, for each $\langle\omega,\hat{A}\rangle$ pair we execute 10^6 simulations. In each simulation we create a new RGG, randomly select a source-destination pair (s,d) and record the number of message copies received by d. For each combination of $\omega,\hat{A},H,|\mathcal{N}(d)|$, we construct the statistical pmf based on the frequency observed for each value of X. Let $\widetilde{\mathrm{P}}\{X=k\}$ denote this pmf. We approximate ϕ in (2) by solving ϕ+$(1-\phi)(1-\omega)^{|\mathcal{N}(d)|}=\widetilde{\mathrm{P}}\{X$=$0\}$. Then, we calculate the total variation distance (d_{TV}) [38] between (2) and $\widetilde{\mathrm{P}}\{X$=$k\}$, i.e. the maximum difference between the probabilities assigned by the two distributions to the same event.

Table II reports d_{TV} values for networks with $\hat{A}$=$\{4,6,8\}$ and N=100. The lower value of $\hat{A}$ corresponds to relatively dense networks while the highest has been chosen so that the resulting networks are as sparse as possible but not partitioned with high probability [39]. We have obtained similar results for various values of N, however, for brevity, we report only the results for $N=100$. According to the presented results, (2) provides a satisfactory approximation for the purposes of the following analysis. As a final note, (2) can be generalized to include the case of transmission errors, i.e. when $\rho\neq 0$. Simulation results (omitted for brevity) confirm that the approximation is still good if ω is replaced by $\omega(1-\rho)$. Furthermore, we have also obtained results confirming that (2) is still valid when nodes' positions follow the random waypoint distribution [35]. This is in accordance with a similar observation regarding the bimodal behavior of probabilistic broadcasting under the same node distribution [37].

TABLE II
TOTAL VARIATION DISTANCE ($\times 10^{-2}$) OF THE APPROX. DISTRIBUTION

	$\hat{A}=4$				$\hat{A}=6$					$\hat{A}=8$				
ω	$\lvert\mathcal{N}(d)\rvert$	$H=2$	$H=3$	$H=5$	$\lvert\mathcal{N}(d)\rvert$	$H=2$	$H=4$	$H=6$	$H=8$	$\lvert\mathcal{N}(d)\rvert$	$H=2$	$H=4$	$H=7$	$H=9$
0.9		0.99	1.21	0.90		0.93	0.82	1.49	0.37		0.77	1.58	2.06	1.35
0.7		0.56	0.84	1.32		0.68	1.00	0.51	1.01		2.24	2.16	1.13	0.43
0.5	6	1.28	1.61	1.40	4	2.74	1.40	0.82	0.60	2	1.44	0.39	0.03	0.02
0.3		1.08	1.99	1.72		1.29	0.22	0.07	0.01		0.17	0.06	0.05	0.01
0.9		0.42	0.12	0.73		0.39	0.17	0.13	0.69		0.75	0.75	0.78	1.46
0.7		0.29	0.45	0.47		1.02	0.91	0.58	0.66		2.04	1.58	1.01	1.01
0.5	14	0.62	0.33	0.62	8	2.72	1.68	1.29	0.73	4	1.81	1.12	0.62	0.27
0.3		2.37	1.88	1.14		3.45	0.92	0.20	0.15		1.01	0.13	0.02	0.02
0.9		0.66	0.58	1.78		0.61	0.95	0.99	1.79		1.10	0.88	0.45	1.16
0.7		0.64	0.22	1.64		0.92	0.73	0.46	0.78		2.63	1.41	1.02	0.58
0.5	22	0.52	0.47	2.20	12	1.42	1.49	1.58	1.40	7	3.57	1.83	0.30	0.34
0.3		2.07	1.92	1.37		5.20	1.25	0.48	0.21		2.89	0.37	0.08	0.02

C. Delivery Efficiency

The performance of RLNC depends on the ability of a node to fully or partially decode a generation, which in turn depends on the rank of the decoding matrix. We examine the usual approach in the literature, in which a source node transmits a new encoded packet each time a native packet is created and added in a generation. In the context of RLNC, each intermediate node, instead of forwarding a received encoded packet, creates a new one. As a result, each node will receive a number of encoded packets with a probability given by (2). The received encoded packets may increase the rank of the decoding matrix, depending on whether they are innovative or not. We assume that the delay from a source to a receiver is smaller than the time between the creation of two native packets, so that all the encoded packets, created after adding the $(k-1)$-th native packet, arrive before the ones created after adding the k-th. Then, the rank of the decoding matrix can be modeled as a stochastic process $Z=\{Z_k, k\in\mathbb{N}\}$, where the RV Z_k denotes the rank after a node d receives all the encoded packets created by the k-th native packet. Note that Z is memoryless because Z_k depends only on the total number of innovative packets received after k–1 native packets (i.e. Z_{k-1}). Therefore, Z is a discrete-time Markov chain and its state space is $[0,g]\in\mathbb{N}$. In the following, we focus on analysing the best case performance of RLNC in order to illustrate its full potential for providing increased delivery efficiency. We discuss the case of non-optimal performance in Section IV-D. Suppose that, at time $k-1$, the rank of the decoding matrix of node d is i, i.e. $Z_{k-1}=i$, and that d

[1] Observe that X can be seen as a set of Y i.i.d. Bernoulli RVs. Then, $G_X(z)=G_Y(G_B(z))$, where G denotes the probability generating function and B indicates a Bernoulli RV

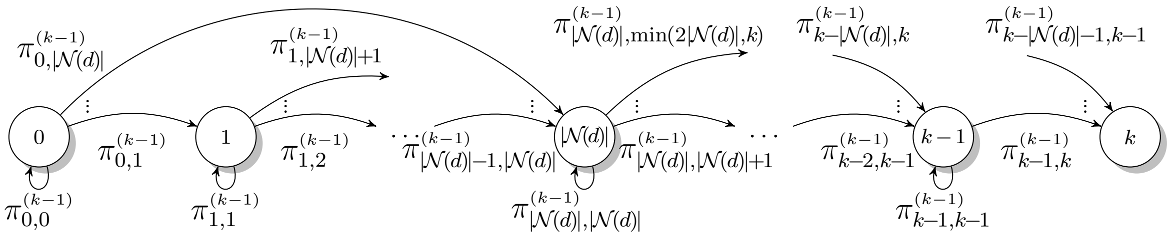


Fig. 1. Proposed Markov chain at time $(k-1,k], k>|\mathcal{N}(d)|$.

receives $j-i \le |\mathcal{N}(d)|$ encoded packets. If n_{in} denotes the number of the encoded packets that are also innovative, then $Z_k = i + n_{in}$. Note that $n_{in} \le j-i$. Furthermore, $Z_k \le k$ because at time k only k native packets have been added in the generation. This implies that $n_{in} \le k-i$. Therefore, $n_{in} \le \min\{j-i, k-i\}$. The best performance occurs when the rank of the matrix is maximal or, equivalently, n_{in} is maximal. When $j<k$, the best performance is when $n_{in}=j-i$, i.e. all the received encoded packets are also innovative. In this case, $Z_k = i+(i-j)=j$ and the transition probability from state i to state j is therefore equal to the probability of receiving $j-i$ encoded packets. However, when $j \ge k$, only $k-i$ out of the $j-i$ encoded packets are innovative because Z_k cannot exceed k. In this case, $Z_k = i+(k-i)=k$ and the transition probability from state i to state k is equal to the probability of receiving $k-i$ or more encoded packets. Summarizing, the transition probabilities in the interval $(k-1,k]$, $1 \le k \le g$ are:

$$\pi_{i,j}^{(k-1)} = \begin{cases} \mathrm{P}\{X=j-i\} & j-i \le |\mathcal{N}(d)|, j<k, i<k \\ \sum\limits_{w=k-i}^{|\mathcal{N}(d)|} \mathrm{P}\{X=w\} & j-i \le |\mathcal{N}(d)|, j=k, i<k \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

For $k>g$, $\pi_{i,i}^{(k-1)} = 1$ and $\pi_{i,j}^{(k-1)} = 0, \forall j \ne i$ since after time $k=g$ no native packets are added in the generation. Note that the Markov chain is time-inhomogeneous. Fig. 1 illustrates the transition probabilities for the time interval $(k-1,k], k>|\mathcal{N}(d)|$. The initial distribution is $\mathrm{P}\{Z_0=0\}=1$ and $\mathrm{P}\{Z_0=i\}=0, \forall i>0$. Therefore:

$$\mathrm{P}\{Z_k=i\} = \sum_{w=0}^{g} p_{w,i}^{(k)} \mathrm{P}\{Z_0=w\} = p_{0,i}^{(k)} \quad (4)$$

where $p_{0,i}^{(k)}$ is the element of table $\boldsymbol{\Pi}^{(k)} = \boldsymbol{\pi}^{(0)}\boldsymbol{\pi}^{(1)} \cdots \boldsymbol{\pi}^{(k-1)}$ in the position $(0,i)$ and $\boldsymbol{\pi}$ are the transition matrices constructed using (3). Decoding is possible when $Z_k=k$ because k innovative packets are required for decoding the k native packets that exist in a generation at time k[2]. Furthermore, decoding of exactly k packets occurs when $Z_k=k$ but no further decoding is possible, i.e. $Z_w<w, \forall w>k$. As a result, the expected delivery rate is:

$$D_R = \frac{\sum\limits_{k=1}^{g} \left[k\mathrm{P}\{Z_k=k\} \prod\limits_{w=k+1}^{g} (1-\mathrm{P}\{Z_w=w\}) \right]}{g} \quad (5)$$

where $\prod_{w=k+1}^{g}(1-\mathrm{P}\{Z_w=w\})$ is the probability that no decoding is possible for $w>k$.

[2]We underestimate the decoding probability since decoding may be possible even if $Z_k<k$

In XOR-based coding, packets are encoded under the requirement that each recipient node will decode at maximum one native packet. In other words, receiving an encoded packet is equivalent to receiving a copy of a native packet. Since receiving a single copy is enough, the expected delivery rate is $D_X = 1-\mathrm{P}\{X=0\}$. Note that, this is the best case performance as we do not take into account decoding failures.

D. Probabilistic broadcasting considered harmful

Fig. 2(a) illustrates the expected delivery rate for RLNC and XOR-based coding when combined with flooding ($\omega=1$). More specifically, the delivery rate is plotted versus the node degree using different values of ϕ and ρ. RLNC exhibits high levels of resilience to transmission impairments and dominates XOR-based coding even when ϕ and ρ increase. A simple explanation is that using RLNC in broadcasting enables a node to exploit message redundancy (or equivalently path diversity) to recover not just a single packet but any packet from the generation. This is possible through the creation of new encoded packets in each intermediate node, which allows a node to receive a plethora of possibly useful packets. This also explains why RLNC fails when message redundancy is absent ($|\mathcal{N}(d)|=1$). RLNC-based schemes should specially treat such cases. As a final note, the proposed Markov chain can be easily generalized to describe the cases that a node receives less than the maximum number of innovative packets per native one. Our results indicate that RLNC continues to dominate the best-case performance of XOR-based coding (for a wide range of ϕ and ρ values). The only exception is when a node receives, at maximum, only one innovative packet for each native, i.e. again when diversity is absent. Nevertheless, such a situation is highly unlikely.

The advantage of XOR-based schemes is the reduced cost, since coding is utilized towards reducing transmissions. Instead, RLNC-based schemes resort to probabilistic forwarding for reducing cost. Fig. 2(b) depicts the performance of RLNC

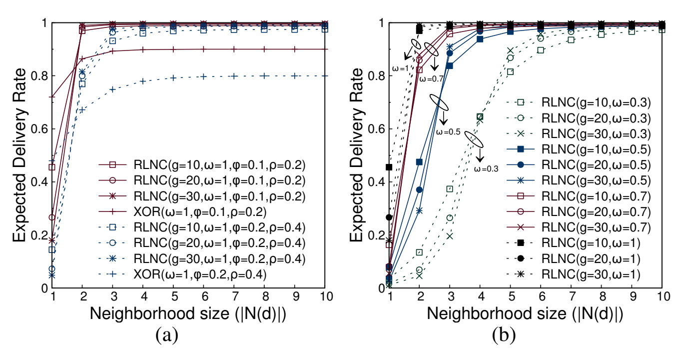


Fig. 2. Analysis of RLNC's delivery efficiency: (a) comparison with XOR-based coding, (b) impact of probabilistic forwarding (ϕ=0.1,ρ=0.2).

when combined with probabilistic forwarding for different values of ω. The plotted results are in accordance with other reported simulation data [5]. Clearly, pruning transmissions significantly impairs RLNC's performance. This performance degradation has also been identified, implicitly [4] or explicitly [10], however the problem has been treated within the context of probabilistic broadcasting. We believe that the key factor for RLNC's performance degradation is the unsystematic way of pruning transmissions, which does not take into account information about connectivity. The strategy to prune transmissions based on heuristics that account for the node degree [4] is towards the correct direction. However, we feel that such heuristics should also take into account topology-related information of non-local scope. Some of this information is difficult to obtain and even if this was possible, it would require a complex analytical model to define the optimal ω. Therefore, *we opt for a more systematic and self-configuring pruning mechanism that takes into account the network topology*.

V. The Synergy of RLNC and Deterministic Broadcasting

Following the previous observations, we adopt RLNC. Yet, contrary to the common approach, we implement it on top of a deterministic broadcast algorithm. We choose Partial Dominant Pruning (PDP) [28] from the class of Dominant Pruning (DP) algorithms. DP algorithms distributively construct a CDS in order to broadcast messages. Our intuition is that the CDS will provide a topology-aware, self-configuring process for reducing transmissions. However, establishing this synergy, without damaging RLNC's coding efficiency, is not a trivial task. PDP's forwarding rules need to be redesigned so as to treat packets as members of a group, i.e. the generation. *Random Linear network coding over Dominant Pruning (RLDP)* incarnates the aforementioned concepts.

A. Basic Concepts

1) Dominant Pruning fundamentals: In DP algorithms, a node v, with a message to broadcast, decides which of its neighbors should act as forwarders and informs them by piggybacking on the message the corresponding list, called the *forwarding set* ($fs(v)$). The process is then repeated by every forwarder until a *termination criterion* is met [28]. Forwarders should be elected so as to deliver the message to (or "cover" according to the set cover terminology) the set of nodes that lie exactly 2-hops away from v. This latter set is also called the universal set $\mathcal{U}(v)$, i.e. $\mathcal{U}(v) = \mathcal{N}(\mathcal{N}(v)) - \mathcal{N}(v)$, where $\mathcal{N}(\mathcal{N}(v))$ is the set of nodes lying within 2-hops from v. The set of candidate forwarders $\mathcal{C}(v)$ consists of v's neighbors, i.e. $\mathcal{C}(v)=\mathcal{N}(v)$. Note that $\mathcal{U}(v)\subseteq\bigcup_{\forall u\in\mathcal{C}(v)}(\mathcal{N}(u)–\mathcal{N}(v))$ and that $\mathcal{C}(v)$ can be seen as a set of sets if each node $u \in \mathcal{C}(v)$ is replaced by $\mathcal{N}(u)–\mathcal{N}(v)$, thus the set cover problem. The problem is solved using the well-known greedy set cover (GSC) algorithm [40], however other approximation algorithms exist [41], [42]. PDP makes the observation that, when v receives a message from u, both $\mathcal{C}(v)$ and $\mathcal{U}(v)$ can be reduced by eliminating the nodes covered by u, i.e. $\mathcal{C}(v)=\mathcal{N}(v) - \mathcal{N}(u)$ and $\mathcal{U}(v)=\mathcal{N}(\mathcal{N}(v))–\mathcal{N}(v)–\mathcal{N}(u)–\mathcal{N}(\mathcal{N}(u) \cap \mathcal{N}(v))$.

2) RLNC related concepts in broadcasting: Broadcasting is inherently coupled with some degree of message redundancy, i.e. a node receives multiple copies of a message, due to path diversity. RLNC takes advantage of this property to enhance error resilience. The idea is to allow forwarders to create new random linear combinations of the generation contents so that a node receives many different encoded packets. Proposed algorithms [4]–[6] use RLNC and build a generation using packets from the same source (intra-source coding) or from different sources (inter-source coding). In the context of RLNC-based broadcasting, inter-source coding operates on an end-to-end basis similar to the intra-source one, i.e. packets are linearly encoded at the source, re-encoded at intermediate nodes and only decoded at the destinations after traveling multiple hops. The only difference is in the composition of the generation. Therefore, this implementation of inter-source coding is also oriented towards resilience to transmission failures. The idea of mixing packets from different sources/sessions is not new and has been extensively used in the literature of network coding based unicast routing [43]. In this line of research inter-source coding is also used for enhancing multi-hop communication but it works on a hop-by-hop basis, i.e. requires decoding and re-encoding at each hop. This also applies to hybrid approaches that combine intra- and inter-source coding [44]–[48], where the former works on an end-to-end basis while the latter on a hop-by-hop one. This hop-by-hop coding approach is reasonable because mixing packets from different flows (i.e. traveling between different source/destination pairs) is meaningful at flow intersection points. Packets from different flows should then be decoupled in the next (probably non common) hop in order to be delivered to the different destinations. On the contrary, in broadcasting packets from different sources are destined to every node in the network, i.e. they share the same set of destinations. Therefore coding and decoding of packets can be performed at the communication end points (i.e. in an end-to-end fashion).

B. Coding Rules

As mentioned previously, RLDP utilizes the basic functionality of random linear coding. Below, we discuss some important design choices and the rationale behind them.

Strictly Inter-source coding: RLDP adopts *inter-source coding* in the light of empirical evidence which demonstrates that it increases the coding efficiency when combined with intra-source coding compared to the case that only intra-source coding is used [6]. However, in this case the problem of generation management is not trivial. In contrast to the usual approach, which is to operate inter-source and intra-source coding in parallel [4]–[6], in RLDP, each source can add only one packet in each generation. In other words, we adopt inter-source coding but do not allow intra-source coding. We call this strategy *strictly inter-source coding (SIS).* Our approach stems from the observation that, in the context of multi-source energy efficient broadcasting, intra-source coding may pose performance issues for poorly connected source-receiver pairs. To understand this recall that the performance of

RLNC degrades under poor connectivity because of the limited message redundancy experienced by a receiver. In inter-source coding however, various sources enroll in a generation. While some sources may form poorly connected pairs with a given destination, it is possible that the opposite holds for some other. This latter set of sources can compensate for the limited redundancy provided by the former one, thus increasing the decoding probability. In other words, inter-source coding exploits what we call "spatial diversity", i.e. the fact that sources located in various parts of the network can provide different levels of message redundancy to a specific destination. Clearly, when the number of sources reduces, the coding gains of SIS deteriorate because it is more probable that a destination will experience poor connectivity with all the sources. In the case of intra-source coding, the only way to overcome the problem of a poorly connected source-destination pair is to employ "*temporal redundancy*", i.e. to allow each intermediate node to subsequently transmit more coded packets. Although this is an efficient approach for unicast scenarios, in the case of broadcasting it results in rapidly increasing the overall number of transmissions because a non trivial percentage of the nodes in the network act as forwarders. Besides the apparent impact on the energy cost, our observation is that this approach may also significantly increase the end-to-end delay. The reason is the elevated number of collisions that can delay the decoding of a generation. Consequently, we have chosen to rule out intra-source coding and focus on SIS. We validate the effectiveness of our approach in various settings (Section VII and Section VIII), including scenarios with limited number of sources. Of course, in the extreme case of a single source the only way to enjoy the benefits of coding is to use the intra-source approach. A node can use its decoding matrix to detect such cases and switch to a strictly intra-source operation. In the rest of the paper we do not examine this scenario since we focus on the multi-source case. As a final note, adopting SIS also allows us to simplify the generation management, i.e. make it easier for source nodes to collectively agree on the grouping of packets into generations. We discuss generation management in detail in Section VI.

Encoding: Similar to every RLNC scheme, each node maintains a decoding matrix for each known generation. A generation is considered known if the node either created it or has received at least one encoded packet belonging to it. After creating a new native packet, the source node either starts a new generation or chooses from the set of known ones in order to add the packet (we discuss in detail this issue in Section VI). Then, it immediately creates and transmits a new encoded packet. The rationale of this strategy is twofold; first it aims to ensure that the new information carried by the native packet will be propagated through the network with minimum delay. Second, it facilitates partial decoding, which reduces end-to-end delay. To understand this, bear in mind that non zero rows of a decoding matrix correspond to innovative packets while non zero columns correspond to native packets. Consequently, the strategy of immediately transmitting a new encoded packet, increases the probability that the decoding matrix contains a full rank square submatrix, thus enabling partial decoding.

Decoding: As mentioned previously, a node can attempt to perform partial decoding instead of waiting for the decoding matrix to become full rank. Deleting the decoding matrix and the packets of a generation is an important decision for managing storage. A frequently used practice is to employ feedback information that allows a receiver to indicate that it has successfully decode a generation, e.g. [15], [10]. However, such approaches require an extremely large number of control messages in the context of broadcasting. This is because all nodes act as receivers and most of them are involved in the forwarding process. Proposed techniques for reducing control messages, e.g. using local scope advertisements [10], [16], are not suitable for mobile networks. A more flexible approach is to allow a node to define a time threshold after which a generation is deleted. The threshold represents the maximum acceptable delay for receiving a packet and can be adjusted to also take into account each node's storage profile.

Coding during Forwarding: Besides being a receiver, a node may be required to act as an intermediate and forward an encoded packet after receiving an innovative one. In that case, the node forwards a new encoded packet. The latter can be created using the packets received up to that time without the need to decode the native ones. In the following, we delineate the conditions under which a node should act as a forwarder.

C. Forwarding Rules

1) Propagating generations over the CDS: In order to achieve the synergy of RLNC and PDP, we need to enable the *propagation of generations through the CDS* formed by PDP. Note that, in DP algorithms, a node v reacts to the reception of a packet only if it has been selected as a forwarding node. Furthermore, recall that, in RLNC, only a subset of the encoded packets, the innovative ones, carry useful information about a generation. Therefore, the first intuitive approach is to adopt the following forwarding strategy:

Definition 1 (Innovative-based criterion): A forwarding node produces and transmits a new encoded packet iff it receives an innovative packet.

In the context of DP, the Innovative-based criterion is actually a termination criterion, i.e. the execution of the algorithm stops when a non innovative packet is received. This criterion is the analogous of the stopping conditions adopted by schemes that implement RLNC on top of probabilistic broadcasting [4], [6]. Given the Innovative-based criterion, we can prove the correctness of RLDP[3], i.e. that all network nodes can fully decode a generation in a lossless network. First, we prove that:

Lemma 1: Every node v, which is not the source node of a native packet q, receives *at least* one innovative packet after q is added in a generation.

Proof: The source node s, after adding q to a generation i, defines a forwarding set $fs(s)$=$\{f_1, f_2, \ldots\}$ and transmits a new encoded packet $e_{s,i}$. Every node $v \in \mathcal{N}(s)$ will receive this packet, which is innovative since it "contains" q. Furthermore, the solution of the set cover problem guarantees that, given a node $u \in \mathcal{N}(\mathcal{N}(s))$, there is at least one forwarder $f \in fs(s)$

[3]We assume that the probability of producing linearly dependent encoded packets is negligible [32]. This assumption is common in the related literature.

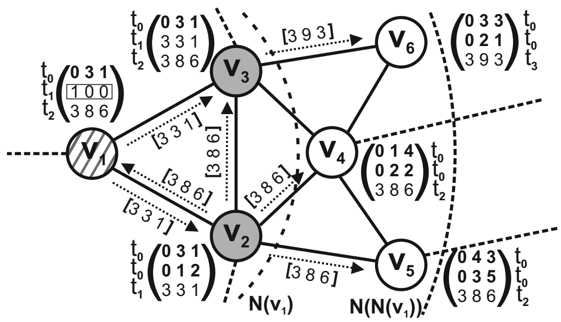


Fig. 3. Example of broadcasting with RLDP (Single-Innovative criterion).

RLDP (prev_node u, cur_node v, packet p, generation_id gid)

```
if (!Innovative(p)) then
    DropPacket(p)
end if
UpdateDecodingMatrix(p)
if (v ∉ p.forwarders || !Single-innovative(p.src, gid) then
    DropPacket(p)
end if
newp=RandomLinearCoding(gid)
fwset=GSC(N(v), N(N(v)), u)
newp.set(fwset)
transmit newp
```

Fig. 4. Pseudocode of RLDP's forwarding procedure.

that covers u. Since $f \in \mathcal{N}(s)$, it will receive the innovative packet $e_{s,i}$ and will transmit a new encoded packet $e_{f,i}$. As a result, each node $u \in \mathcal{N}(\mathcal{N}(s))$ will receive at least one encoded packet $e_{f,i}$. The first of these packets is clearly innovative since it "contains" q. The same reasoning can be used in subsequent hops to include all network nodes. ∎

Moreover, we can prove that:

Lemma 2: Every node v, which is not the source node of a native packet q, receives *exactly* one innovative packet after q is added in a generation.

Proof: According to Lemma 1, if g native packets are added in generation i, then each node v will receive $g' \geq g$ innovative packets. It suffices to show that $g'=g$. Note that the row rank of $\mathbb{G}_{v,i}$ (which equals g') cannot exceed the column rank (which equals g), i.e. $g'>g$ is not possible. ∎

We use this lemma to prove that:

Theorem 1 (Correctness of RLDP): Every node can decode a generation in a lossless network.

Proof: Lemma 2 secures that, any non source node v will receive exactly g innovative packets for a generation of size g, thus $\mathbb{G}_{v,i}$ has a full rank. Furthermore, each source s will receive exactly $g-1$ innovative packets, one for each native packet added by the other sources. This is sufficient since s only needs to decode g–1 packets. ∎

2) Reducing transmissions: According to Lemma 2, the Innovative-based criterion is equivalent to the strategy of forwarding one encoded packet for each native one added in a generation. However, in the presence of transmission errors, if a native packet is added in a generation, a node v will receive more than one innovative packet. This happens when the rank of its decoding matrix is lower than the rank of the decoding matrices of its neighbors. Using the Innovative-based criterion in such cases will result in v transmitting more than one encoded packet for each native one. To explain the situation, let us examine the example in Fig. 3. In this example, we monitor the decoding matrices ($g=3$) in a part of a network. At some point in time (t_0), the generation contains already two native packets (added by some other nodes in the network). As a result, each node has received at most two innovative packets and populated its matrix accordingly (entries marked with t_0). Note that, due to transmission errors, v_1 and v_3 have received only one innovative packet. At some point, v_1 acts as a source, adds a packet in the generation and after selecting the forwarding set (in this case $fs(v_1)=\{v_2, v_3\}$), transmits an encoded packet (the transmitted innovative packets are illustrated with dashed lines along with the corresponding encoding vectors). Both v_2 and v_3 receive an innovative packet and update their matrices (entries marked with t_1). Then, v_2 (a forwarder that received an innovative packet) transmits an encoded packet which is received by v_3, v_4 and v_5 (entries marked with t_2). Note that v_3 receives two innovative packets (one from v_1 and one from v_2) and, according to the Innovative-based criterion, should transmit two encoded packets. We make the observation that, in the context of Dominant Pruning, *not all innovative packets need to result in the transmission of a new encoded packet*. In fact, we introduce the following policy:

Definition 2 (Single-innovative criterion): A forwarding node produces and transmits a new encoded packet only for the first innovative packet that is received as a result of the addition of a native packet in a generation.

The rationale of this policy is clear and, in part, is expressed by Lemma 2 and Theorem 1; in the absence of transmission errors only one innovative packet per native is adequate while, in the presence of transmission errors, *a node should only rely on the path diversity provided by the network to recover from transmission errors*. To further explain, let us go back to the example of Fig. 3. When the Single-innovative criterion is used, v_3 receives two innovative packets and decodes the generation. However, v_3 will transmit only one new encoded packet. Note that this new packet is enough for v_6 to decode the generation (entry marked with t_3). Furthermore, observe that v_6 actually receives two encoded packets (the second one is from v_4 and is not illustrated since it is not innovative). If the rank of v_6's decoding matrix was initially one, both of the received packets would be innovative. Therefore, v_6 could take advantage of path diversity and decode the generation. Clearly, there is still the probability that a node will not be able to decode a generation. In general, this probability increases for nodes with low connectivity. One solution to eliminate failures would be to allow a node to relax the Single-innovative criterion based on the connectivity of its neighbors or even based on feedback information. We examine such a solution in Section VIII. However, for now, we refrain from investigating the impact of such methods, as well as the related cost, since our primary objective is to illustrate that using deterministic broadcasting, even without such methods, results in less decoding failures compared to a probabilistic scheme.

An important issue is how to implement the Single-innovative criterion. To do so, we need to provide some kind of association between a native packet q and the innovative ones produced after node s adds q in a generation i. Since in RLDP a node adds only one native packet into a generation, this task can be tackled by using the value pair $\langle s, i \rangle$, i.e. the source address and the generation id, which is contained in a packet's header. Another requirement is to allow a forwarding node v to track whether an innovative packet with the same value pair $\langle s, i \rangle$ has already been received. The most efficient way is to use direct addressing [40] due to the fast dictionary operations. The space complexity of such an approach ($\mathcal{O}(g)$ for a generation of size g) is reasonable since the generation size is usually kept low in order to reduce the decoding cost. Fig. 4 presents the pseudocode of RLDP's forwarding procedure.

VI. Distributed Generation Management

In order to practically implement RLNC, packets should be grouped into generations. This task, known as *generation management*, is easier to handle when intra-source coding is used since the required decisions involve a single node and thus are made locally. In this case, the main challenge is to select the generation size g so that it maximizes a chosen performance feature (see e.g. [49]). However, generation management becomes more complicated in the case of inter-source coding since the sources should agree on a common grouping of packets in a distributed fashion. In the following, we focus on the challenges that arise from this requirement:

1) Decide which generation to choose for adding a native packet and when to start a new generation: Choosing a generation is the first important decision to make because it affects the overall performance. Observe that, when inter-source coding is used, a node v may be unaware of the existence of a generation or have incomplete view of the number of packets added in it. The reason is that when another node u adds a packet to an existing generation or starts a new one, node v becomes aware of this after receiving an encoded packet produced from this generation. Therefore, the challenge is to ensure that the number of packets added in each generation will be close to the predefined size g. Let $\mathcal{GS}_v$ denote the set of generations which are known to v (i.e. v has received at least one encoded packet from or created the generation) and their size, according to v's view, has not exceeded the size g. The common approach is that a source s will add a new packet to a randomly chosen generation from $\mathcal{GS}_s$ [4], [5]. Another approach is to choose from a subset of $\mathcal{GS}_s$ which contains generations initiated from nodes that lie certain hops away from s [4], [6]. A new generation is started if $\mathcal{GS}_s = \emptyset$ [4]–[6] or when the chosen generation already contains a packet from s [5]. All the proposed strategies aim at reaching the predefined generation size, in order to increase performance [5]. However, under transmission errors, a large generation size increases the decoding delay. The reason is that it takes longer to collect the number of encoded packets that is required for decoding. Following this observation, we opt for reduced delay. Therefore, in RLDP, a source s adds a new packet to the most recently seen generation, if this belongs to $\mathcal{GS}_s$ (see the pseudocode in Fig. 5 that presents the process for selecting a generation). If no such generation is found or the selected generation already contains a packet from s (strictly inter-source coding), then a new generation is created. Note

GetGeneration (set $\mathcal{GS}_s$, generation_id $last\_seen$)

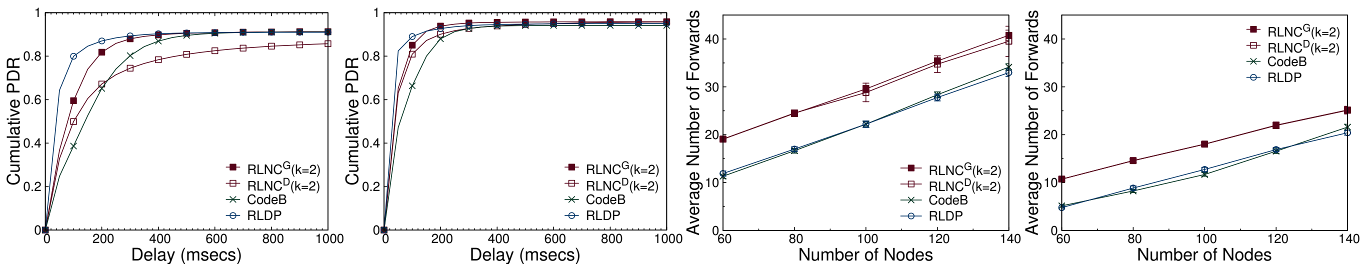
```
if (last_seen ∈ GS_s) then
    if (AlreadyUsed(last_seen) == TRUE) then
        last_seen = last_seen + 1    //create a new generation
    end if
else
    last_seen = last_seen + 1    //create a new generation
end if
return last_seen
```

Fig. 5. Pseudocode for selecting a generation in RLDP.

that, the size of the produced generations will not necessarily be close to g. However, we believe that this will not have a significant impact on the decoding efficiency. This intuition is based on reported empirical data [4], [5], also confirmed by the analysis in Section III, which indicate that a relatively small generation size is enough for providing the coding benefits. We confirm our intuition through simulation in Section VII.

2) Provide an addressing scheme for packets within a generation: Another problem, although rarely discussed in the literature, is to uniquely identify packets within a generation. To understand this requirement, recall that each encoded packet carries an encoding vector, i.e. the coefficients $\langle c_1, \ldots, c_g \rangle$ used to mix the native packets. In order for decoding to be possible, it is necessary that all nodes will be able to agree on and use the same mapping between native packets and coefficients. This is a challenging task in a distributed environment. A practical solution is to provide a unique id for each native packet, so as to enable sorting based on this id, and associate it with the corresponding coefficient. The simplest way to accomplish this is by using the pair $\langle node\_id, seq\_num \rangle$, where seq_num is a sequence number generated locally at the source and $node\_id$ is the source address [6]. The use of seq_num enables two packets from the same source to coexist in a generation. RLDP takes a simpler approach. Since strictly inter-source coding is used, there is no way that two native packets from the same source will reside in the same generation. Therefore, only $node\_id$ can be used for uniquely identifying a packet in the generation. Our strategy, besides using a smaller identifier for packets, does not involve any overhead for managing sequence numbers.

3) Provide an addressing scheme for generations: The next important task is to uniquely identify generations so that each node can decide to which generation an encoded packet belongs to. The problem arises when a source, due to the incomplete knowledge of existing generations, uses an id to start a new generation without knowing that this has already been used by another source. Consequently, a node may receive two encoded packets with the same generation id but constructed using different native packets. The downside is that it is possible to destroy the one-to-one mapping between the native packet ids and the coding coefficients. To tackle the problem, the usual approach is that a node will randomly choose the generation id [4]–[6]. Choosing from a sufficient large space minimizes the probability that two different nodes will choose the same generation id. In RLDP however, two sources can use the same generation id without destroying the aforementioned mapping. This is because, in any case, a generation will contain, at maximum, only one native packet per source and this will be uniquely identified by $node\_id$.



(a) (b) (c) (d)

Fig. 6. Performance for different network densities (λ=1 pkt/sec/source, max speed:1 m/sec): (a) Cumulative PDR vs delay ("Sparse", N=100) (b) Cumulative PDR vs delay ("Dense", N=100) (c) Avg. number of forwards vs number of nodes ("Sparse") (d) Avg. number of forwards vs number of nodes ("Dense").

Therefore, when a source starts a new generation, increases by one the most recently seen generation id and uses this as the new id (Fig. 5). This strategy allows sources to coordinate their views about the generations in use, thus enabling them to effectively populate the generations with packets. Eliminating the random selection of generation ids allows the use of a smaller address space.

VII. Evaluation of RLDP

To evaluate RLDP's performance, we compare it with two algorithms. The first one, proposed in [4], is the most representative of RLNC-based algorithms. In the following, we will use the term RLNC to refer to this algorithm. The second algorithm, CodeB [7], utilizes XOR-based coding. Regarding RLNC, we use two variants, namely RLNC^D and RLNC^G. The first, uses the distributed generation management described in [4]. In the second, we assume that each node has global coding information, i.e. perfect knowledge of the coding status of other nodes. This scheme achieves the optimal allocation of packets across generations. Although it is unrealistic, we use it to illustrate the performance bounds of RLNC. Furthermore, RLNC employs the forwarding heuristic described in [4, Algorithm 6B] with k=2. We chose this setting after extensive experimentation which showed that it yields the best performance in our experiments, i.e. it results in the best possible trade-off between delivery efficiency and the number of forwards.

Set up and methodology: All investigated algorithms are implemented in the ns2 simulator [50], using the CMU extension. Furthermore, RLNC and RLDP were implemented based on the network coding ns2 module [51]. We present the average values over 20 independent simulation runs, each with a duration of 900 seconds. The confidence level, for all reported confidence intervals, is 95%.

Network model: The default number of nodes is 100, the propagation model is the TwoRay ground with a transmission range of 250m and the nominal bit rate is 2Mbps. The nodes move in a square area according to the Random Waypoint (RW) model [35]. To avoid transient artifacts in nodes' movement, we use the perfect simulation algorithm [52]. We examine two network densities; "Dense" and "Sparse". Similar to [7], in the "Dense" topology, the average neighborhood size is 30 while in the "Sparse" topology it is 15. Note that, we could not use a lower density in the "Sparse" scenario since, in such a case, frequent partitions would occur. Simulations showed that in the 'Sparse" scenario, there exist many nodes (those moving near the boundaries) that experience very low connectivity. All algorithms collect neighborhood information by periodically exchanging hello messages with an interval of 1 second.

Network traffic: Traffic is generated by broadcast sessions, each stemming from a different source node and starting at a random time. The size of each message is set to 256 bytes. Furthermore, both the number of sources and the maximum generation size are fixed to 30. We chose the generation size after extensive experimentation, which showed that using a larger size does not improve performance but rather increases the related costs. We used a GF of size 2^8.

Fig. 6(a) and 6(b) depict the cumulative packet delivery ratio (PDR) versus the end-to-end delay, i.e. the cumulative fraction of native packets received by a node within a delay limit, for "Sparse" and "Dense" networks. We only consider decodable packets for calculating PDR. Moreover, for the calculation of a packet's end-to-end delay we use the time instant that decoding of this packet becomes feasible. However, we do not consider the decoding delay because it depends on the implementation specifics of each scheme, thus making it impossible for a fair comparison. We choose the aforementioned presentation style in order to capture both the delivery efficiency and the timeliness of each algorithm. The results provide a confirmation of the effectiveness of random linear network coding. Both RLDP and RLNC^G outperform CodeB. The main reason is that XOR-based coding schemes introduce delay in order to detect coding opportunities. As expected, in the "Sparse" topology, the performance of all schemes degrades. For CodeB, a low density topology reduces the coding opportunities. As a result, more transmissions occur (compare Fig. 6(c) and 6(d)) and increase the probability of collisions. In the case of random linear coding, the witnessed degradation is in accordance to the analysis in Section IV because in low density topologies the average neighborhood size is smaller. Nonetheless, RLDP outperforms both RLNC^D and RLNC^G, which uses global knowledge. This justifies our approach to combine random linear coding with deterministic broadcasting. Note that, in sparse topologies, RLNC^D fails to keep up with other schemes. This highlights the importance of distributed generation management. Also, observe that, RLDP's generation management does not compromise the coding gains. We tested networks of various sizes (from 60 to 140 nodes) and found qualitatively similar results. Fig. 6(c) and 6(d) illustrate the average number of forwards versus the network size for "Sparse" and "Dense" networks. The results confirm the intuition that the CDS, used by RLDP to forward messages, provides an efficient pruning process.

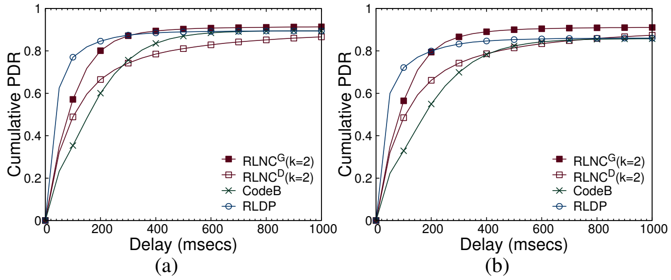


Fig. 7. Cumulative PDR vs delay (λ=1 pkt/sec/source, "Sparse" topology). Node speed: (a) $2-10$ m/sec (b) $10-20$ m/sec.

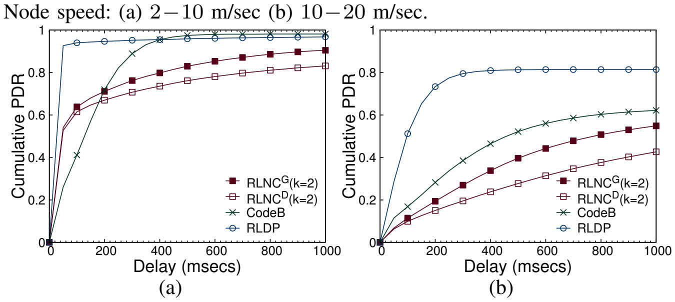


Fig. 8. Cumulative PDR vs delay (max speed:1 m/sec, "Sparse" topology): (a) $\lambda=0.1$ (b) $\lambda=2$ pkts/sec/source.

More specifically, RLDP manages a reduction from 17% to 38% in "Sparse" and from 19% to 56% in "Dense" networks, compared to RLNC variants. Interestingly, RLDP performs similar to CodeB, despite the fact that the latter uses coding for reducing transmissions.

In the following experiments, we only examine the delivery efficiency since we observed similar findings as far as the number of forwards is concerned. Furthermore, we focus on the more challenging scenario of "Sparse" networks. Fig. 7 presents the delivery efficiency under different levels of mobility. Clearly, increased mobility levels impacts the performance of RLDP and CodeB. The reason is that both schemes use deterministic broadcasting, which is affected by topology variations. Moreover, mobility also increases the decoding failures in CodeB since successful decoding depends on the accuracy of information about the neighbors' coding status. On the contrary, RLDP minimizes the impact of mobility on the deterministic broadcasting algorithm due to the use of random linear coding. Both RLNC^D and RLNC^G are virtually unaffected by mobility. This is attributed to the higher message redundancy produced by the probabilistic forwarding scheme. Nevertheless, message redundancy results in a significantly increased cost (more than 30% compared to RLDP) in terms of transmissions. In any case, observe that only the unrealistic RLNC^G outperforms RLDP when mobility is very high.

In Fig. 8, we evaluate the algorithms for different levels of traffic load. Under low traffic (Fig. 8(a)), RLDP outperforms all algorithms. CodeB needs to wait an increased amount of time in order to find coding opportunities since fewer packets coincide in the network. Both RLNC variants suffer from increased delay as they need more time to fill the generations. On the other hand, RLDP outperforms all schemes because its generation management is oriented towards reducing delay. The tradeoff is a reduced number of packets allocated to each generation (refer to Section VI). However, this does not impair the delivery efficiency. When congestion levels increase (Fig. 8(b)), the performance of all algorithms degrades. However, RLDP exhibits a remarkable resilience due to the combination of deterministic broadcasting and random linear coding. The former reduces the levels of congestion and thus decreases the probability of collisions. The latter uses path diversity to enhance delivery efficiency. Both mechanisms are equally important. CodeB and RLNC variants fail because they use only one of them.

VIII. Coping with Poorly Connected Nodes

The analysis in Section IV has demonstrated that RLNC's performance deteriorates in poorly connected nodes, i.e. nodes with a small number of neighbors (or equivalently limited message redundancy), even in the absence of probabilistic forwarding (refer to Fig. 2(a) where $\omega=1$). The phenomenon intensifies as the probability of transmission failures ρ increases. Therefore, it becomes evident that *we should enhance message redundancy in a topology-aware fashion*, i.e. increase the number of encoded packets received by poorly connected nodes. Before implementing such a strategy, we make two important observations. The first is that, in RLDP, the message redundancy experienced by a node v depends on the number of forwarders that are located in $\mathcal{N}(v)$ (let $|\mathcal{N}^{\mathrm{F}}(v)|$ denote this number). This is in contrast to gossip-based forwarding, where the message redundancy depends on $|\mathcal{N}(v)|$. Consequently, a node v for which $|\mathcal{N}^{\mathrm{F}}(v)|$ is small is considered as poorly connected. The second important observation is that only forwarding nodes can enhance message redundancy because they are the only ones that transmit. Thus, any action for enhancing message redundancy should be taken by forwarders.

To tackle the aforementioned problem, the first important question is whether a forwarding node should try to enhance the message redundancy experienced by its neighbors or not. Our approach is that this should be done by a forwarding node that is a:

Definition 3 (Border Forwarder): A forwarding node that is the only one to cover one or more network nodes.

Equivalently, a forwarding node f_i is a border forwarder when there exists at least one neighbor v for which f_i is the only forwarder in $\mathcal{N}(v)$, i.e. $|\mathcal{N}^{\mathrm{F}}(v)|=1$. More formally, $f_i \in fs(u)$, where $fs(u)$ is the forwarding set constructed by node u, is a border node iff:

$$\exists\, v \in (\mathcal{N}(f_i)-\mathcal{N}(u)) : v \notin \mathcal{N}(f_j), \forall f_j \neq f_i \in fs(u)$$

The rationale of using border forwarders is straightforward; a border forwarder should take action because it is responsible for delivering the message to at least one poorly connected

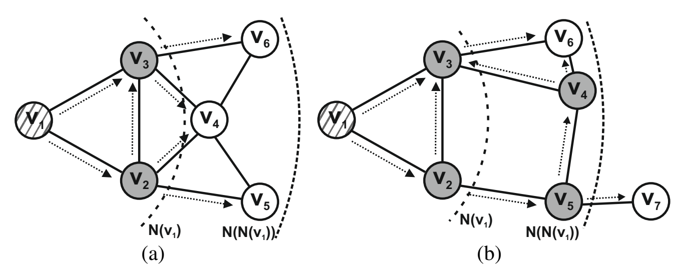


Fig. 9. Example of a node (v_3): a) correctly and b) falsely identifying itself as a border forwarder.

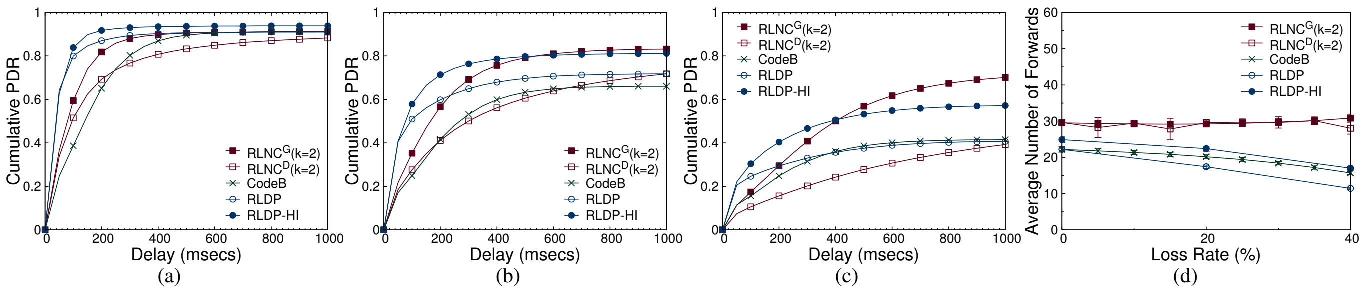


Fig. 10. Performance for different loss rates ("Sparse", N=100, λ=1 pkt/sec/source, max speed:1 m/sec): (a) Cumulative PDR vs delay (Loss rate 0%) (b) Cumulative PDR vs delay (Loss rate 20%) (c) Cumulative PDR vs delay (Loss rate 40%) (d) Avg. number of forwards vs loss rate.

node v because $|\mathcal{N}^{\mathrm{F}}(v)| = 1$. The advantage of this approach is that, at the same time, there is a high probability that other nodes, located in the neighborhood of the border forwarder, are also experiencing relatively poor connectivity and could benefit from its actions. Let us examine the example in Fig. 9(a). Node v_6 is only covered by v_3, therefore v_3 is a border forwarder. At the same time, node v_4, although covered by two forwarders (v_3 and v_2), is relatively poorly connected, i.e. $|\mathcal{N}^{\mathrm{F}}(v_4)| = 2$. Consequently, both v_6 and v_4 could benefit if v_3 decides to act for enhancing message redundancy.

In order for a forwarding node f_i to identify itself as a border forwarder, it suffices to check whether there exists a node $v \in (\mathcal{N}(f_i) - \mathcal{N}(u))$ such that:

$$\forall f_j \neq f_i \in fs(u), f_j \notin \mathcal{N}(v)$$

Note that all required information for performing this test, i.e. $fs(u)$, $\mathcal{N}(u)$ and $\mathcal{N}(v), \forall v \neq u \in \mathcal{N}(f_i)$ is available to f_i. However, this test is based on local scope information, therefore f_i may falsely identify itself as a border forwarder.

To illustrate this, let us examine the example in Fig. 9(b). Again, v_3 identifies itself as a border forwarder although v_6 is now covered by two forwarders (v_3 and v_4). This happens because v_3 is not aware of the fact that v_5, at a later time, chooses v_4 as a forwarder. Nonetheless, we do not wish to eliminate the occurrence of such events. The reason is that, although a false decision does not indicate a node covered by a single forwarder, it is highly correlated with the identification of poorly connected nodes, which are also in need of enhanced message redundancy. For example, although v_6 is not anymore covered only by v_3, it is still a poorly connected node.

After deciding which forwarding node should act, the next important decision is what policy it should implement to increase the message redundancy. A first simple approach would be to transmit multiple encoded packets each time the Single-Innovative criterion is activated, i.e. when the first innovative packet is received as a result of the addition of a native packet in a generation. The main drawback of this approach is that it is not straightforward how to derive the appropriate number of encoded packets. Therefore, we take a more elaborate approach. We let border forwarders to relax the Single-Innovative criterion and instead implement the Innovative-based one. More specifically, we introduce the following termination criterion:

Definition 4 (Hybrid-Innovative criterion): A forwarding node implements the Innovative-based termination criterion if it is a border forwarder and the Single-Innovative criterion in any other case.

There are two advantages in this approach. The first is that it is topology-aware due to the use of border forwarders. The second is that it can adapt to network conditions. When the probability of transmission failures is small most packets are delivered and the rank of a node's decoding matrix is close to the maximum. Thus, the number of innovative packets decreases and transmissions are suppressed. On the other hand, when loss rate is high, the rank of many nodes falls behind the maximum possible rank, therefore more packets are innovative. As a result, this termination criterion will result in more packets being sent to poorly connected nodes.

A. *Evaluation of RLPD-HI*

To evaluate the new termination criterion and compare its performance to the other algorithms, we experiment by introducing transmission failures in the channel. More specifically, we use the error model in ns2 [50]. The model defines a loss rate l as the result of channel impairments. When a node transmits a message, each of its neighbors receives the message with a probability $1 - l$. We tested the algorithms for values of l from 0 to as high as 0.4. Note that l only captures the packet losses due to channel impairments while ρ refers to all packet losses, including those owned to collisions or even stale neighborhood information. Therefore, $\rho \geq l$ and $l = 0$ does not imply that packets losses do not occur. In the experiment, we vary l because the packet losses due to collisions and stale neighborhood information depend on on traffic levels and node mobility, respectively, thus it is impossible to quantify ρ.

Fig. 10(a)-10(c) illustrate the cumulative PDR vs the end-to-end delay for all schemes and for various values of l. We use RLDP-HI to denote RLDP with the Hybrid-Innovative termination criterion. The results confirm our approach. Introducing the Hybrid-Innovative criterion to RLDP improves the performance even when $l = 0$ (recall that even in this case $\rho \geq 0$, so there is room for improvement). RLDP-HI presents a remarkable resilience to transmission failures and it is outperformed only by RLNC^G and only when l is as high as 0.4. The latter is a reasonable result since RLNC^G features the unrealistic scenario of perfect knowledge about generations. On the other hand, all fully distributed schemes experience a higher performance degradation because the increased loss rate has an impact on the accuracy of a node's local information about generations. In any case, RLDP-HI outperforms all other fully distributed schemes, including CodeB. Although CodeB does not use packet generations, its performance declines for different reasons. Recall that,

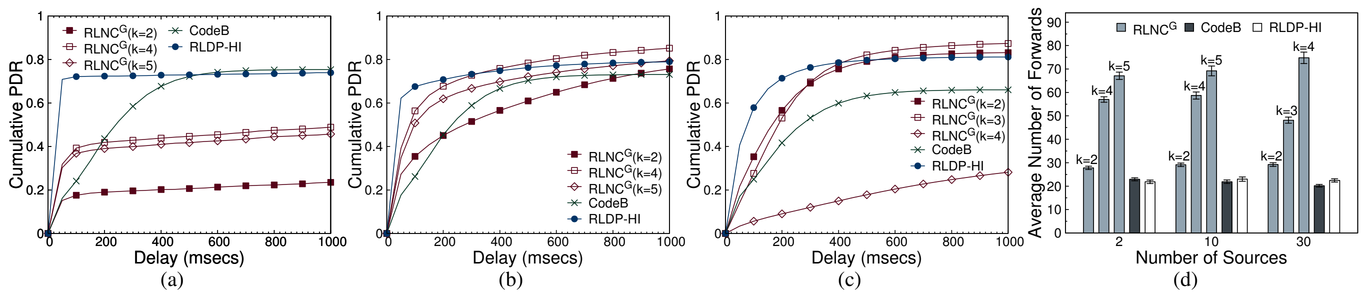


Fig. 11. Performance for different source count(λ=1 pkt/sec/source, max speed:1 m/sec, N=100, Loss Rate 20%, "Sparse" topology): (a) Cumulative PDR vs delay (2 sources) , (b) Cumulative PDR vs delay (10 sources), (c) Cumulative PDR vs delay (30 sources) and (d) Avg. number of forwards.

in CodeB, a node v maintains information about the packets received by another node u (previously denoted as B_u^v) in order to identify coding opportunities and secure that successful decoding is possible. Nonetheless, packet losses significantly invalidate the information in B_u^v, thus leading to decoding failures.

As expected, introducing the Hybrid-Innovative termination criterion results in more transmissions (Fig. 10(d)). Nevertheless, the increase is minimal, proving the efficiency of the criterion. This result is more impressive if we bear in mind that the number of transmissions and the delivery rate are correlated; dropping a packet aborts future transmissions, thus creating a bias in favour of the other schemes. Indeed, RLDP-HI performs close to CodeB, in terms of transmissions, but at the same time improves the delivery rate by $\sim 3\%$ when $l = 0$ and $\sim 17\%$ when $l = 0.4$. Besides being efficient due to its topology-awareness, the Hybrid-Innovative criterion also presents a remarkable adaptability to the loss rate, i.e. it performs equally well, in terms of transmissions, for small and high loss rates. This is a confirmation of the rationale that led us to the introduction of the Hybrid-Innovative termination criterion. Moreover, this performance characteristic renders RLDP-HI as the best solution regardless of the loss rate.

Next, we tested all schemes by changing the number of sources that are present in the network. As discussed in Section V-B, the source count affects the performance of RLDP. Obviously, the impact is more severe in the presence of losses since message redundancy deteriorates. Therefore, we have chosen to present the results in the presence of channel loss rate ($l = 0.2$) and found analogous results for values of $l \in [0, 0.4]$. Fig. 11 presents the performance of RLDP-HI, RLNC^G and CodeB when there exist 2, 10 or 30 sources in the network. We also present the performance of RLNC^G for various values of k. As expected, the performance of RLDP-HI, in terms of the cumulative PDR vs the end-to-end delay (Fig. 11(a)-11(c)), degrades as the number of sources decreases. This is reasonable because less packets are included in a generation and therefore there are less opportunities to exploit the "spatial diversity" that we discussed about in Section V-B. However, the degradation is limited since less traffic results in less packet collisions. Interestingly enough, a similar, but more severe, performance degradation is witnessed for RLNC^G regardless of the value of k. Although part of this degradation (mostly in the case of 2 and 10 broadcasting sources) can be attributed to the time required for filling a generation, the major reason is related to intra-source coding and the use of temporal redundancy. Recall that RLNC^G implements both inter- and intra-source coding. When the number of sources decreases, the coding process resembles a pure intra-source approach since more packets from the same source are included in a generation. As a result, the role of "temporal redundancy" (i.e. the ability of an intermediate node to transmit more encoded packets), which is necessary for coping with losses in this case, becomes more critical in decoding a generation. However, in the context of broadcasting, the "temporal redundancy" comes at the cost of delay, thus the performance degradation. To explain this, observe that in broadcasting many forwarders may find themselves within each other's transmitting range. Consequently, there is an increased probability of collisions which can delay the decoding of a generation because a destination may need to wait for subsequent transmissions in order to receive the required amount of encoded packets. Reasonably, increasing the "temporal redundancy" (through k) improves the performance of RLNC^G but there are two downsides. The first is that there is a limit for k after which no improvement is possible and the performance actually degrades (compare for example $k=4$ and $k=5$ in Fig. 11(a)). The reason is that the impact of collision-related failures increases to a level that invalidates the benefits of redundancy. The second and more critical disadvantage is that any performance improvement comes at the expense of a surge in cost (Fig. 11(d)). In the case of CodeB, the performance actually degrades when the number of sources increases. At first, this seems surprising since packets from more sources provide more coding opportunities. However, the higher traffic load increases the collision-related transmission failures. Besides the fact that XOR-based schemes are not error resilient, transmission failures also invalidate the information used to make coding decisions, thus leading to decoding failures. Finally, note that, RLDP-HI either outperforms all schemes or it performs close to RLNC^G but with much less cost although the latter is rather unrealistic and features a much better performance compared to the more realistic RLNC^D. However, we chose RLNC^G in order to focus on the performance characteristics of intra-source coding and rule out other factors related to the distributed implementation.

Finally, we conducted a set of experiments with the presence of channel fading using the well-known Rayleigh model. The model is appropriate for environments with many obstacles that block the line-of-sight between the transmitter and the receiver. In this experiment we do not introduce errors using an error model. Instead, transmission failures occur due to

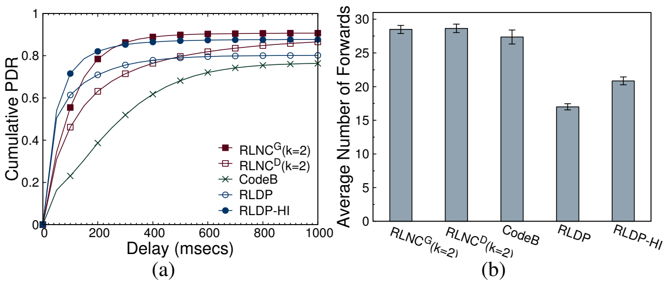


Fig. 12. Performance of all schemes under Rayleigh fading ($\lambda = 1$ pkt/sec/source, max speed:1 m/sec, N=100, "Sparse" topology): a) Cumulative PDR vs delay, (b) Avg. number of forwards

fading and are more frequent as the distance between the communicating nodes increases. Fig. 12(a) depicts the cumulative PDR vs the end-to-end delay. The performance of all schemes degrades compared to the case that there is no fading (Fig. 10(a)). The reasons are the same as those explained in the experiment with the uniform error model. Still, RLDP-HI outperforms all distributed schemes while its performance is comparable to that of RLNC^G. CodeB experiences a notable increase of the average number of forwards (Fig. 12(b)). This is because the effective transmission range is smaller than 250m since more distant nodes experience very poor link quality. As a result, the underlying PDP algorithm uses more forwarders to cover the same network area. Although both RLDP and RLDP-HI also rely on PDP, their efficient termination criteria allow them to suppress transmissions and outperform both RLNC and CodeB.

IX. Conclusion

Random linear network coding is used to enhance the resilience of protocols to packet losses. We proved, through analysis, that we need to utilize a topology-aware algorithm in order to maximize its benefits. To this end, despite the common approach in the literature, which is to use random linear coding on top of probabilistic forwarding schemes, we chose the synergy with a CDS-based broadcast algorithm. Furthermore, we proposed an extension of the basic algorithm in order to enhance topology-awareness and cater for poorly connected nodes, especially when the packet loss rate is high. We demonstrated, through simulation, the efficiency of both approaches. Moreover, we provided a distributed mechanism for managing generations. The mechanism does not compromise the coding efficiency even in cases of high mobility and increased packet loss rate. In the future, we plan to investigate new generation management techniques and explore their impact on the performance of random linear network coding.

References

[1] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad hoc Netw.*, vol. 2, no. 1, pp. 1–22, 2004.

[2] A. N. Mian, R. Baldoni, and R. Beraldi, "A survey of service discovery protocols in multihop mobile ad hoc networks," *IEEE Pervasive Comput.*, vol. 8, no. 1, pp. 66–74, 2009.

[3] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[4] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "Efficient broadcasting using network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 450–463, 2008.

[5] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proc. ACM SIGCOMM Workshop Delay-tolerant Netw. (WDTN)*, 2005, pp. 284–291.

[6] T. Kunz, K. Mahmood, and A. Matrawy, "Broadcasting in multihop wireless networks: the case for multi-source network coding," in *Proc. IEEE Int. Conf. on Commun. (ICC)*, 2012, pp. 6679–6684.

[7] L. Li, R. Ramjee, M. Buddhikot, and S. Miller, "Network coding-based broadcast in mobile ad-hoc networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2007, pp. 1739–1747.

[8] S. Yang and J. Wu, "Efficient broadcasting using network coding and directional antennas in manets," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 21, no. 2, pp. 148–161, 2010.

[9] N. Kadi and K. Al-Agha, "Optimized MPR-based flooding in wireless ad hoc network using network coding," in *Wireless Days*, 2008, pp. 1–5.

[10] A. Asterjadhi, E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "Toward network coding-based protocols for data broadcasting in wireless ad hoc networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 2, pp. 662–673, 2010.

[11] I. Hou, Y. Tsai, T. Abdelzaher, and I. Gupta, "Adapcode: Adaptive network coding for code updates in wireless sensor networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2008, pp. 1517–1525.

[12] S. Y. Cho and C. Adjih, "Wireless broadcast with network coding: Dynamic rate selection," in *Proc. IFIP Annu. Med. Ad Hoc Netw. Workshop*, 2008, pp. 191–202.

[13] Z. Yang, M. Li, and W. Lou, "R-Code: Network coding-based reliable broadcast in wireless mesh networks," *Ad Hoc Networks*, vol. 9, no. 5, pp. 788–798, 2011.

[14] J. Subramanian, R. Morris, and H. Balakrishnan, "UFlood: High-throughput flooding over wireless mesh networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2012, pp. 82–90.

[15] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proc. ACM SIGCOMM*, 2007, pp. 169–180.

[16] D. Koutsonikolas, Y. C. Hu, and C.-C. Wang, "Pacifier: High-throughput, reliable multicast without "crying babies" in wireless mesh networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 5, pp. 1375–1388, 2012.

[17] S. Wang, G. Tan, Y. Liu, H. Jiang, and T. He, "Coding opportunity aware backbone metrics for broadcast in wireless networks," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 25, no. 8, pp. 1999–2009, Aug 2014.

[18] N. Kadi and K. Agha, "Network coding for MPR-based flooding in ad hoc wireless networks under mobility conditions," *Springer-Verlag Ann. Telecommun.*, vol. 66, no. 5-6, pp. 319–329, 2011.

[19] N. Rahnavard and F. Fekri, "CRBcast: A collaborative rateless scheme for reliable and energy-efficient broadcasting in wireless sensor networks," in *Proc. ACM Inf. Process. Sensor Netw.*, 2006, pp. 276–283.

[20] B. Vellambi, N. Rahnavard, and F. Fekri, "FTS: A distributed energy-efficient broadcasting scheme using fountain codes for multihop wireless networks," *IEEE Trans. Commun.*, vol. 58, no. 12, pp. 3561–3572, 2010.

[21] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *ACM/IEEE Proc. Int. Conf. Mobile Comput. and Netw.*, 1999, pp. 151–162.

[22] C. Ververidis and G. Polyzos, "Service discovery for mobile ad hoc networks: a survey of issues and techniques," *IEEE Commun. Surveys Tuts.*, vol. 10, no. 3, pp. 30–45, 2008.

[23] Y. Luo and O. Wolfson, "Mobile p2p databases," in *Springer Encyclopedia GIS*, 2008, pp. 671–677.

[24] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. Allerton Conf. Commun., Control, and Comput.*, 2003, pp. 40–49.

[25] S.-Y. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. on Inf. Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[26] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. IEEE Int. Symp. Inf. Theory*, 2003, p. 442.

[27] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, pp. 243–254, 2006.

[28] W. Lou and J. Wu, "On reducing broadcast redundancy in ad hoc wireless networks," *IEEE Trans. Mobile Comput.*, vol. 1, no. 2, pp. 111–122, 2002.

[29] P. Ostovari, A. Khreishah, and J. Wu, "Deadline-aware broadcasting in wireless networks with network coding," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2012, pp. 4435–4440.

[30] M. Luby, "Lt codes," in *Proc. Annu. IEEE Symp. Found. Comput. Sci.*, 2002, pp. 271–280.

[31] P. Ostovari, J. Wu, and A. Khreishah, "Deadline-aware broadcasting in wireless networks with local network coding," in *Proc. IEEE Int. Conf. Comput. Netw. and Commun. (ICNC)*, 2012, pp. 1123–1127.

[32] Y. Wu, P. Chou, and K. Jain, "A comparison of network coding and tree packing," in *Proc. IEEE Int. Symp. Inf. Theory*, 2004, p. 143.

[33] M. Penrose, *Random geometric graphs (Oxford Studies in Probability)*. Oxford University Press Oxford, 2003, vol. 5.

[34] P. Nain, D. Towsley, B. Liu, and Z. Liu, "Properties of random direction models," in *Proc. IEEE INFOCOM*, 2005, pp. 1897–1907.

[35] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 2, no. 3, pp. 257–269, 2003.

[36] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-based models of delivery and interference in static wireless networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, pp. 51–62, Aug. 2006.

[37] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Trans. Netw.*, vol. 14, no. 3, pp. 479–491, 2006.

[38] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*. Amer. Math. Soc., 2006.

[39] M. J. Appel and R. P. Russo, "The connectivity of a graph on uniform points on $[0, 1]^d$," *Statist. & Probability Lett.*, vol. 60, no. 4, pp. 351–357, 2002.

[40] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*. McGraw-Hill Higher Education, 2001.

[41] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," in *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, 2002, pp. 3866–3875.

[42] S. Agathos and E. Papapetrou, "On the set cover problem for broadcasting in wireless ad hoc networks," *IEEE Commun. Lett*, vol. 17, no. 11, pp. 2192–2195, 2013.

[43] L. Xie, P. H. Chong, I. W. Ho, and Y. Guan, "A survey of inter-flow network coding in wireless mesh networks with unicast traffic," *Elsevier Computer Networks*, vol. 91, pp. 738 – 751, 2015.

[44] H. Seferoglu, A. Markopoulou, and K. Ramakrishnan, "I^2NC: Intra- and inter-session network coding for unicast flows in wireless networks," in *IEEE Proc. INFOCOM*, 2011, pp. 1035–1043.

[45] C. Qin, Y. Xian, C. Gray, N. Santhapuri, and S. Nelakuditi, "I^2MIX: Integration of intra-flow and inter-flow wireless network coding," in *IEEE Proc. Annu. Conf. Sensor, Mesh Ad Hoc Commun. Netw. Workshops (SECON)*, 2008, pp. 1–6.

[46] X. Zhu, H. Yue, Y. Fang, and Y. Wang, "A batched network coding scheme for wireless networks," *Springer Wireless Networks*, vol. 15, no. 8, pp. 1152–1164, 2009.

[47] A. Khreishah, I. Khalil, and J. Wu, "Low complexity and provably efficient algorithm for joint inter and intrasession network coding in wireless networks," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 24, no. 10, pp. 2015–2024, 2013.

[48] J. Krigslund, J. Hansen, M. Hundeboll, D. E. Lucani, and F. H. P. Fitzek, "CORE: COPE with MORE in wireless meshed networks," in *IEEE Veh. Technol. Conf. (VTC)*, 2013, pp. 1–6.

[49] J. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros, "Network coding meets tcp: Theory and implementation," *IEEE Proc.*, vol. 99, no. 3, pp. 490–512, 2011.

[50] S. McCanne and S. Floyd. ns Network Simulator. [Online]. Available: http://www.isi.edu/nsnam/ns/

[51] ns2 simulation code for wireless network coding. [Online]. Available: http://telecom.dei.unipd.it/simulation

[52] J.-Y. Le Boudec and M. Vojnovic, "Perfect simulation and stationarity of a class of mobility models," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2005, pp. 2743–2754.