Technical Report
LCA-REPORT-2005-009
July 2005

A Network Coding Approach to Energy Efficient Broadcasting: *from Theory to Practice*

Christina Fragouli
EPFL
Lausanne, Switzerland
Email: christina.fragouli@epfl.ch

Jörg Widmer[1]
DoCoMo Euro-Labs
Munich, Germany
Email: widmer@docomolab-euro.com

Jean-Yves Le Boudec
EPFL
Lausanne, Switzerland
Email: jean-yves.leboudec@epfl.ch

***Abstract*— We show that network coding allows to realize significant energy savings in a wireless ad-hoc network, when each node of the network is a source that wants to transmit information to all other nodes. Energy efficiency directly affects battery life and thus is a critical design parameter for wireless ad-hoc networks. We propose an implementable method for performing network coding in such a setting. We analyze theoretical cases in detail, and use the insights gained to propose a practical, fully distributed method for realistic wireless ad-hoc scenarios. We address practical issues such as setting the forwarding factor, managing generations, impact of transmission range and mobility. We use theoretical analysis and packet level simulation.**

I. Introduction

Network coding is an area that has emerged in 2000 [1], [2], and has since then attracted an increasing interest, as it promises to have a significant impact in both the theory and practice of networks. We can broadly define network coding as allowing intermediate nodes in a network to not only forward but also process the incoming information flows. Combining independent information flows allows to better tailor the information flow to the network environment and accommodate the demands of specific traffic patterns.

The first paradigm that illustrated the usefulness of network coding established throughput benefits when multicasting over error-free links. Today, we have realized that we can get benefits not only in terms of throughput, but also in terms of complexity, scalability, and security. These benefits are possible not only in the case of multicasting, but also for other network traffic configurations, such as multiple unicast communications. Moreover, they are not restricted to error-free communication networks, but can also be applied to sensor networks, peer-to-peer systems, and optical networks.

In this paper we show that use of network coding allows to realize significant energy savings when broadcasting in wireless ad-hoc networks. By broadcasting we refer to the problem where each node is a source that wants to transmit information to all other nodes. Such one-to-all communication is traditionally used during discovery phases, for example by routing protocols; more recently, it has been described as a key mechanism for application layer communication in intermittently connected ad-hoc networks [3].

Energy efficiency directly affects battery life and thus is a critical design parameter for wireless ad-hoc networks. Optimizing broadcasting for energy efficiency has been extensively studied during the last decade. Since flooding results in a prohibitively large overhead [4], a substantial number of more efficient algorithms have been proposed. The problem of minimum energy broadcasting in ad-hoc wireless networks is NP-complete [5] and a large number of approximation algorithms exist. Usually, these are either based on probabilistic algorithms (see for example [4], [6], [7]) where packets are only forwarded with a certain probability, or some form of topology control (e.g., [8], [9], [10]) to form connected dominating sets of forwarding nodes.

The new ingredient in this problem is that we can apply ideas from the area of network coding. Use of network coding has been examined in the literature in conjunction with multicasting, when a single source transmits common information to a subset of the nodes of the network. If we allow intermediate nodes to code, the problem of minimizing the energy per bit when multicasting can be formulated as a linear program and thus accepts a polynomial-time solution [11]. An alternative formulation is presented in [12], where a distributed algorithm to select the minimum-energy multicast tree is proposed. Broadcasting information from a single source to all nodes in the network is a special case of multicasting and thus the same results apply. The problem we examine in this paper is further distinct in that all nodes of the network are sources, but for convenience we again simply refer to this problem as broadcasting. In [13] we quantified the energy savings that network coding has the potential to offer when broadcasting in ad-hoc wireless networks. The analysis was over canonical configurations, and assuming perfect centralized protocols. We also presented preliminary simulation results over random networks.

We examine different aspects of the proposed system in detail, that are related to and motivated by practical considerations. The emphasis of the paper is both in understanding

[1]Part of this work was done while Jörg Widmer was with EPFL.

the theoretically expected performance, and in developing algorithms using the insights gained. In particular,

• We theoretically examine benefits in terms of energy efficiency that use of network coding can bring to this problem *without idealized centralized scheduling*, that is, when we restrict our attention to distributed algorithms. Based on this analysis, we propose distributed algorithms that are tuned to random networks.

• We evaluate possible tradeoffs of parameters that arise in a practical systems such as the effect of the transmission range and the effect of mobility.

• We also develop distributed algorithms that can be deployed in real networks; we address fundamental considerations such as the choice of a forwarding factor, and other practical considerations such as restricted complexity and memory capabilities, and limited generation sizes.

We evaluate fundamental tradeoffs and the performance of our algorithms both on systematic networks (circular network and square grid, where we can find exact results) and on random, realistic networks (where we obtain simulation results).

The paper is organized as follows. Section II formally introduces the problem formulation and reviews previous theoretical results. In Section III we present our proposed distributed algorithms. Section IV discusses the effect of changing the transmission range. Section V examines the effect of node mobility. Section VI develops algorithms for constrained complexity and memory requirements, and Section VII concludes the paper.

II. Background Material

In this section we first formally introduce the problem formulation and notation. We then briefly review known results that are related to our approach, and discuss how our work is placed in this framework. We also describe the simulation environment that we will use to evaluate our algorithms.

A. Problem Formulation

Consider a wireless ad hoc network with n nodes, where each node is a source that wants to transmit information to all other nodes. We are interested in the minimum amount of energy required to transmit one unit of information from a source to all receivers.

We assume that each node v can successfully broadcast one unit of information to all neighbors $N(v)$ within a given transmission range, through physical layer broadcast. We also assume that the transmission range is the same for all nodes. Thus, minimizing the energy is equivalent to minimizing the number of transmissions required to convey a unit of information from a source to all receivers.

More precisely, let T_{nc} denote the total number of transmissions required to broadcast one information unit to all nodes when we use network coding. Similarly, let T_w denote the required number of transmissions when we do not use network coding. We are interested in calculating $\frac{T_{nc}}{T_w}$.

Note that the same problem formulation, minimizing the number of channel uses (transmissions) per information unit, can be equivalently viewed as maximizing the throughput when broadcasting. Thus our results can also be interpreted as bounding the throughput benefits that network coding can offer, for our particular traffic and network environment.

Let $x_1, \dots x_n$ denote the source symbols associated with the n nodes. These symbols[1] are over a finite field $\mathcal{F}_q$. Each linear combination y over $\mathcal{F}_q$ that a node transmits or receives, can be described as the product of a vector of coefficients and a vector of source symbols

$$y = ax = [a_1 \dots a_n] \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}. \quad (1)$$

In the network coding literature, the n-dimensional vector of coefficients is referred to as a coding vector. Following the approach in [14] we assume that packets (coded symbols) are always sent together with the corresponding coding vectors.

In the following it will be convenient to think in terms of vector spaces, and say that a node has received a vector space spanned by m coding vectors, when the node has received the m corresponding linear combinations of the source symbols. Each node v stores its source symbol and the information vectors it receives, in a decoding matrix G_v, that contains the tuples of the coding vectors and the received information symbols. The matrix of a source s_i that has not yet received information from any other node contains only a single row (e_i, x_i). A received packet is said to be innovative if its vector increases the rank of the matrix. Reception of non-innovative packets is simply ignored.

In the case of network coding, a node v will in general transmit a linear combination that lies in the vector space of its decoding matrix G_v. We can think of flooding or probabilistic routing (i.e., without the use of network coding) as constraining the coding vectors to belong in the set of the orthonormal basis elements

$$e_1 = [1\ 0\ 0 \dots 0],\ e_2 = [0\ 1\ 0 \dots 0],\ \dots,\ e_n = [0 \dots 0\ 1].$$

Thus in this case G_v is a submatrix of the identity matrix.

Once a node has received n linearly independent combinations, or equivalently, a basis of the n-dimensional space, the node is able to decode and retrieve the information of the n sources. In the case of network coding, decoding amounts to solving a system of linear equations, with complexity bounded as $O(n^3)$. In the case of probabilistic routing no decoding is required.

B. Previous Results

In [13] we evaluated the theoretical energy requirements for broadcasting with and without network coding, over canonical networks, and assuming perfect centralized scheduling. More precisely, we characterized the optimal performance we may

[1] Equivalently, we can think of $x_1, \dots x_n$ as packets of symbols, and apply to each packet the operations symbol-wise. In the following we will talk about symbols and packets interchangeably.

hope to get over these canonical configurations with any transmission scheme, showed that it can be achieved using network coding, and also evaluated what fraction of this optimal value we can achieve using forwarding. For completeness we briefly review these results here.

1. Circular Network: In the circular network n nodes are placed at equal distances around a circle as depicted in Fig. 1.

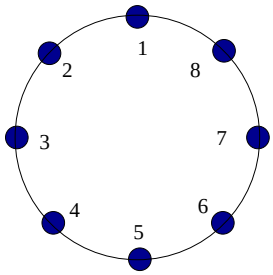

Fig. 1. A circular configuration with 8 nodes.

Assume that each node can successfully broadcast information to its two nearest neighbors. For example, in Fig. 1, node 1 can successfully broadcast information to nodes 2 and 8. In [13] it was shown that

1) without network coding $T_w \geq (n-1)(1+\epsilon)$
2) with network coding $T_{nc} \geq \frac{n-1}{2}(1+\epsilon)$,

where $\lim_{n\to\infty} \epsilon \to 0$. It was also shown that there exist routing and coding schemes that achieve the lower bound, and thus

$$\frac{T_{nc}}{T_w} = \frac{1}{2}. \tag{2}$$

2. Square Grid Network: In this case we consider a wireless ad-hoc network with $n = m^2$ nodes where the nodes are placed on the vertices of a rectangular grid. To avoid edge effects, we will also assume that the area of the grid envelopes the surface of a torus.

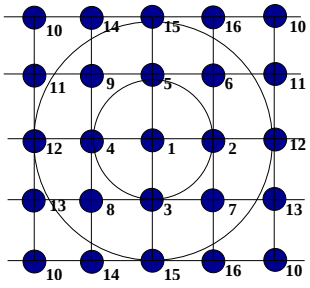

Fig. 2. A rectangular grid configuration. The node numbering expresses the fact that the grid envelopes a torus.

In [13] it was shown that, if each node can successfully broadcast information to its four nearest neighbors then

1) without network coding $T_w \geq \frac{n^2}{3}(1+\epsilon)$
2) with network coding $T_{nc} \geq \frac{n^2}{4}(1+\epsilon)$,

where $\lim_{n\to\infty} \epsilon \to 0$ and that there exist schemes that achieve the lower bounds for T_w and T_{nc}. Thus

$$\frac{T_{nc}}{T_w} = \frac{3}{4}. \tag{3}$$

As is well known, random networks tend asymptotically (in the number of nodes) to behave like square grid networks. We underline that the benefits calculated here refer to an idealized case, where perfect centralized scheduling of all node transmissions is possible. As we will see in the following sections, in more realistic environments, network coding allows to realize significantly larger gains when we constrain both flooding and coding to operate in a distributed manner.

C. Description of the Simulator

Throughout this paper we will verify our theoretical analysis through simulation results over random topologies. Unless explicitly stated otherwise, the simulation environment will be as described in the following.

Nodes have a nominal transmission range of $\rho = 250m$ and are placed on a torus to avoid edge effects. Transmissions are received by all the nodes within transmission range. We use a custom, time-based network simulator. A packet (symbol) transmission takes exactly one time unit. We assume that a node can either send or receive one packet at a time. The MAC layer is an idealized version of IEEE 802.11 with perfect collision avoidance. At each time unit, a schedule is created by randomly picking a node and scheduling its transmission if all of its neighbors are idle. This is repeated until no more nodes are eligible to transmit.

To allow an efficient implementation of network coding, we use operations over the finite field $\mathcal{F}_{2^8}$, so that each symbol of the finite field can be stored in a byte. Addition and multiplication operations over this finite field can be implemented using `xor` and two lookup tables of size 255 bytes [15]. The encoding vectors are transported in the packet header as suggested in [14]. We use randomized network coding, i.e., combine the received vectors uniformly at random to create the vector to transmit.

As performance metrics we mainly use Packet Delivery Ratio (PDR) and decoding delay. The PDR is defined as the number of packets that can be *decoded* at the destination. For probabilistic routing, this is equal to the number of received innovative packets, whereas with network coding, not all innovative packets can necessarily be decoded. Similarly, delay is counted as the average time between the transmission of a packet by the original source and successful decoding at a node, where averaging is across receiver nodes. For some simulations we also investigate total network energy consumption, which is measured as the sum of transmit power $\times$ transmission time for all transmissions over the duration of the simulation.

III. Distributed Algorithms

In this section we are interested in developing distributed algorithms that are well suited for random topologies. To this goal, we first prove that there exists a simple distributed algorithm that uses network coding and allows to achieve the optimal performance over the square grid network. We then tune this algorithm to perform well in a random topology, and verify through simulation that we obtain the expected benefits.

As discussed in Section II, in [13] we proved that there exists a network coding scheme that achieves $T_{nc} = \frac{n^2}{4}(1+\epsilon)$, i.e., the minimum possible number of transmissions. This scheme operates in k iterations, where in each iteration every node collects the information from sources that are at an increased distance from it. Since the number of neighbors depends on the distance, the number of transmissions per node at each iteration is different. The associated scheduling algorithm tends to be involved, since it changes with each time iteration, and thus might be challenging to implement in a practical system.

In the following we will show that there exists a much simpler scheduling that still allows us to achieve the optimal benefits in terms of energy efficiency. The algorithm operates in iterations as follows.

Algorithm 1:

- Iteration 1: Each node broadcasts the information symbol it produces to its four closest neighbors.
- Iteration k: Each node transmits a linear combination of the source symbols that belongs in the span of the coding vectors that the node has received in previous iterations.

A. Theoretical Analysis

Let m_k denote the number of linear independent combinations that node i has received at the end of iteration k, and let V_k^i be the vector space spanned by the corresponding coding vectors. That is, $m_k = |V_k^i|$. Moreover, if A is a set of nodes, denote by V_k^A the union of the vector spaces that the nodes in A span, i.e., $V_k^A = \{\bigcup_j V_k^j,\ j \in A\}$.

To show that Algorithm 1 allows to achieve the optimal performance when broadcasting, we need to show that there exists a coding scheme (linear combinations that nodes can transmit) such that each broadcast transmission brings innovative information to four receivers. This implies that Algorithm 1 operates in $k = 1 \dots \lceil \frac{n}{4} \rceil$ iterations as follows. At iteration k, each node i

1) Transmits a vector from the vector space spanned by the coding vectors the node received at iterations $1 \dots k-1$.
2) Receives four vectors, from his four closest neighbors, and increases the size of his vector space by four.

Before the iterations begin, each node has its own source symbol, and thus $m_0 = 1$. Thus equivalently, it is sufficient to show that for each node i at the end of iteration k

$$m_k = m_{k-1} + 4 = 4k + 1. \tag{4}$$

To prove that there exists a coding scheme such that Eq. (4) holds, it is sufficient to prove that the following theorem holds.

Theorem 1: There exists a coding scheme to be used with Algorithm 1 such that at iteration k,

$$|V_k^A| \geq \min\{m_k + |A| - 1, n\} \tag{5}$$

for **any** set A of nodes in the grid, where $m_k = 4k + 1$, $m_0 = 1$.

Eq. (5) for $A = \{i\}$ gives that $|V_k^i| \geq m_k = 4k+1$. But node i at iteration k has received only $4k$ broadcast transmissions, i.e., $|V_k^i| \leq m_k = 4k + 1$. Thus the theorem directly implies that $|V_k^i| = m_k = 4k + 1$.

For the proof of this theorem, we will use two results, that we describe in Lemmas 1 and 2.

Lemma 1: Any set A of nodes in the grid,with $4+|A| \leq n$, has at least four distinct neighbors.

Proof: The proof uses the fact that the vertex min-cut between any two nodes in a square grid is four. Let B be the set of nodes in the grid that are not in A. From assumption B contains at least four nodes. If all the nodes in B are neighbors of nodes in A we are done. Assume that there exist a node b in B that is not a neighbor of any node in A. Let a be any node in A. Connect a and b through four vertex disjoint paths. On each such path there exists a distinct neighbor of A. ■

The second result we will need, was originally used in the framework of network coding in [16]. Here we write this result in a form that is convenient for the proof of our theorem.

Lemma 2: Consider a family of $n \times n$ matrices $A_1, A_2, \dots A_m$ that are parameterized by coefficients $p_1, p_2, \dots p_l$. Assume that, for each matrix A_i, there exist values $p_1, p_2, \dots p_l$ over a field $\mathcal{F}_{q_i}$ such that the determinant of A_i over $\mathcal{F}_{q_i}$ is non zero, i.e., $det(A_i) \neq 0$. Then, there exists a finite field $\mathcal{F}_q$, and there exist values in $\mathcal{F}_q$ for $p_1, p_2, \dots p_l$ such that $det(A_1) \neq 0$ and $det(A_2) \neq 0 \dots$ and $det(A_m) \neq 0$.

For example, if

$$A_1 = \begin{bmatrix} p^2 & p(1-p) \\ p(1-p) & p^2 \end{bmatrix}, \text{ and } A_2 = \begin{bmatrix} 1 & p \\ 1 & 1 \end{bmatrix}, \tag{6}$$

then for $p = 1$, $det(A_1) \neq 0$ over $\mathcal{F}_2$, for $p = 0$, $det(A_2) \neq 0$ over $\mathcal{F}_2$, and for $p = 2$, both $det(A_1) \neq 0$ and $det(A_2) \neq 0$ over $\mathcal{F}_3$.

Proof of Theorem 1

We will prove this theorem using induction.

• For $k = 0$, $m_0 = 1$, since every node has one source symbol.

• For $k = 1$, $m_1 = 5$. Indeed, at the end of the first iteration each node has received the information symbols from his four nearest neighbors. Selecting any A nodes, we will have the information from the A nodes themselves, and moreover from all their closest neighbors, which, from Lemma 1, will amount to a union vector space of size at least $m_1 + |A| - 1 = |A| + 4$.

• Assume that the condition holds for $k = l-1$. It is sufficient to show that it holds for $k = l$.

Consider a set A. We want to show that $|V_k^A| \geq m_{k-1} + 4 + |A| - 1 = m_k + |A| - 1$. From induction we know that $|V_{k-1}^A| \geq m_{k-1} + |A| - 1$. If $|V_{k-1}^A| \geq m_{k-1} + 4 + |A| - 1$ we are done. The only interesting cases are when $|V_{k-1}^A| = m_{k-1} + i + |A| - 1$, $i = 0 \dots 3$. We will prove here the case where $|V_{k-1}^A| = m_{k-1} + |A| - 1$. For the other three cases the arguments are very similar.

Let B be the set that includes A and all the nearest neighbors of A. From Lemma 1 we know that B contains at least four nodes that do not belong in A, say $\{b_1, b_2, b_3, b_4\}$. We want to show that when the nodes in $\{b_1, b_2, b_3, b_4\}$ transmit during iteration k, they increase the rank of the set A by four. (And in fact, of every other set they are neighbors). But this holds by the following argument. From assumption,

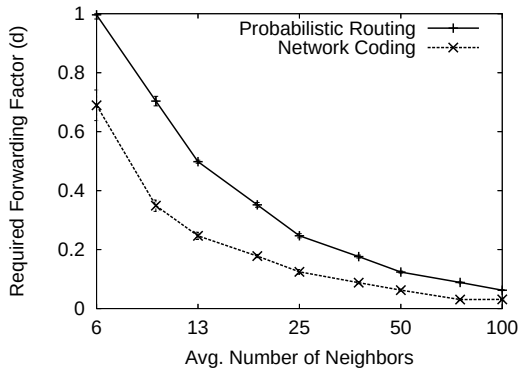

Fig. 3. Forwarding factor required to achieve a 99% PDR for different node densities (in a network of 2000m $\times$ 2000m)

$$|V_{k-1}^{\{A,j\}}| \geq m_{k-1} + |A|, \text{ for } j \in \{b_1, b_2, b_3, b_4\}$$
$$|V_{k-1}^{\{A,j,l\}}| \geq m_{k-1} + |A| + 1, \text{ for } j, l \in \{b_1, b_2, b_3, b_4\}$$
$$|V_{k-1}^{\{A,j,l,z\}}| \geq m_{k-1} + |A| + 2, \text{ for } j, l, z \in \{b_1, b_2, b_3, b_4\}$$
$$|V_{k-1}^{\{A,b_1,b_2,b_3,b_4\}}| \geq m_{k-1} + |A| + 3.$$

Thus, nodes b_1, b_2, b_3 and b_4 have vectors v_1 v_2, v_3 and v_4 respectively such that $v_j \notin V_{k-1}^A$, $j = 1 \dots 4$, and the vector space spanned by them has dimension four, i.e., $| < v_1, v_2, v_3, v_4 > | = 4$. Then, from Lemma 2, there exist linear combinations that nodes b_i can transmit at iteration k such that the vector space of A (and in fact any set A neighboring them) increases in size by four.

To conclude, we have proved that there exists a coding scheme such that the simple distributed scheduling of Algorithm 1 achieves the optimal theoretically performance. In practice we will use randomized coding over a large enough field [17], to approximate this optimal performance.

B. Application to Random Networks

In this section we extend Algorithm 1 to work over random topologies, where the number of neighbors $N(v)$ of a node v is not constant. Moreover, the network is not perfectly symmetric and we cannot assume perfect synchronization among nodes.

To account for these factor, the authors in [13] proposed a network coding protocol in analogy to probabilistic routing algorithms that forwards packets with a certain probability, according to a forwarding factor $d > 0$ [6], [7]. The forwarding factor should intuitively be inversely proportional to the density of a node's neighborhood. In [13] the forwarding factor reflected the average node density of the network. Figure 3 shows which forwarding factor is required to achieve a 90% PDR with probabilistic routing and with network coding. We observe that the overhead of probabilistic routing is higher by a factor of 2-3, except for the case where the node density is so low that a number of nodes have only one or very few neighbors. In this case, network coding as well as probabilistic routing need to use $d = 1$.

In this paper we extend this work by proposing to use a *dynamic forwarding factor*, that is different for each node of the network, and adapts to possible changes of the network topology. The algorithm can be described as follows.

Algorithm 2:

- We associate with each node v in the graph a "forwarding factor" d_v.
- Node v transmits its source symbol $\max\{1, \lfloor d_v \rfloor\}$ times, and an additional time with probability $p = d_v - \max\{1, \lfloor d_v \rfloor\}$ if $p > 0$.
- When a node receives an innovative symbol, it broadcasts a linear combination over the span of the received coding vectors $\lfloor d_v \rfloor$ times, and an additional time with probability $p = d_v - \lfloor d_v \rfloor$ if $p > 0$.

The optimum value of d_v depends on the number of disjoint paths from the information sources to all other nodes and can only be calculated with perfect knowledge of the network topology. Since we are interested in simple distributed algorithms, we assume that a node can acquire knowledge about the direct neighborhood as well as the two-hop neighborhood, while further information is too costly to gather. We will therefore investigate the performance of two simple heuristics to adjust d_v.

Let $N(v)$ be the set of direct neighbors of node v and let k be a forwarding factor to be used when a node only has one single neighbor. We scale d_v as follows:

- *Algorithm 2A:* Set v's forwarding factor inversely proportional to the number of 1-hop neighbors
$$d_v = \frac{k}{|N(v)|}.$$
- *Algorithm 2B:* Set the forwarding factor inversely proportional to the minimum of the number of 1-hop neighbors of v's 1-hop neighbors
$$d_v = \frac{k}{\min_{v' \in N(v)} |N(v')|}.$$

We expect the second scheme to outperform the first. Intuitively, if a node v has multiple neighbors but one of the neighbors v' has only node v as a neighbor, v needs to forward all available information to v', no matter how many neighbors v itself has.

The performance of Algorithm 2B depends on the value of k. In essence, k is a cumulative forwarding factor shared between all nodes within a given radio range. It corresponds to the number of packets that are transmitted *within this coverage area* as a response to the reception of an innovative packet, independent of the node density.

To determine k, we need to compute the probability that a transmitted packet is innovative. In [4], the authors analyze the probability that the broadcast of a given message is innovative for at least one neighbor when this message has already been overheard a certain number of times, for the case of flooding. This probability quickly drops to 0 for more than ca. 6-8 overheard broadcasts of the same message. Therefore, k should be set such that the number of broadcasts in an area is close to this value and independent of the network density.

TABLE I

NUMERICAL VALUES OF Q^g_{kg} (PROBABILITY OF BEING COVERED BY FEWER THAN g OUT OF kg DISKS).

$g=$	1	2	4	$g \to \infty$
$k=1$	0.413	0.636	0.835	1
$k=2$	0.191	0.232	0.261	0.347
$k=3$	0.094	0.0838	0.0635	0
$k=4$	0.0480	0.0304	0.0136	0
$k=5$	0.0252	0.0111	0.00270	0
$k=6$	0.0135	0.00407	0.000505	0

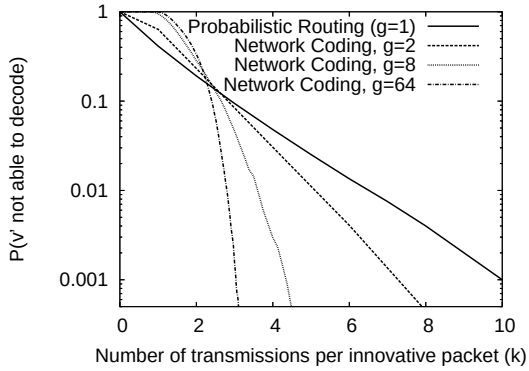

Fig. 4. Probability that a node is not able to decode after kg transmissions for different numbers of information vectors g (i.e., sizes of the decoding matrices).

A similar analysis is possible for network coding. As a rough approximation, let us assume that a node v and all but one of its neighbors have all g information vectors, and one neighbor v' has no information. We are interested in the probability that after overhearing kg transmissions, a packet from v will be innovative for v'. In other words, v' must have received fewer than g innovative packets from the other nodes and is not yet able to decode.[2]

We compute this probability as follows. Let D_0 be a disk of radius 1 (we can take all transmission ranges equal to 1 since the probability we are interested in is independent of the distance unit chosen). Let $j = kg$, and $D_1, ..., D_j$ be j disks, also of radius 1, with centers in D_0, drawn independently and uniformly in D_0. Define Q^g_j as the probability that a random point M in D_0 is covered by fewer than g of the j disks. Our upper bound is the probability Q^g_{kg}. We show in appendix how to compute this in closed form. The results are illustrated in Table III-B. For fixed g and large k, we have the approximation

$$Q^g_{kg} \approx \frac{1.72029}{\sqrt{gk}} e^{-0.321021gk}. \quad (7)$$

The probability of node v's transmission being innovative is depicted in Figure 4 for the case of probabilistic routing ($g = 1$) and network coding ($g > 1$). With probabilistic routing, this probability decreases exponentially with the number of transmissions, while it drops to 0 much more rapidly with network coding. The slope of the curve depends on the number of information vectors g. In the network scenarios we are interested in, g is on the order of tens to hundreds of information vectors. To achieve a reasonably small probability of not being able to decode below 1%, we have to set $k \approx 3$ for network coding and $k > 6$ for flooding. (Note that this is the probability that v' is not able to decode only using transmissions from nodes in $N(v)$. It might still receive packets via some other neighbors, resulting in a lower overall PDR.) Interestingly, for $k \geq 3$, the probability of not being able to decode tends to 0 in the limit for large g, while it is strictly positive for smaller k.

The performance of Algorithms 2A and 2B in random networks coincides well with the above analysis. As expected, the actual forwarding factors that are necessary are slightly lower than indicated by the worst case analysis. In Figure 5, network coding achieves a PDR close to 100% for $k \geq 2$ (except for very low node densities where the network is only partially connected). Probabilistic forwarding requires $k \geq 6$ for a similar performance and thus incurs a per packet overhead that is larger by a factor of 3. The performance for low k increases dramatically with Algorithm 2B, where the 2-hop neighborhood is taken into account (Figure 6). For network coding, the PDR with $k = 1$ increases roughly tenfold, and for k slightly smaller than 1 (e.g. 1.2, not shown in the graph), network coding achieves a PDR 100%. Probabilistic routing benefits as well, however not to the same degree as network coding. For PDRs close to 100%, it still requires $k \geq 6$.

In all of the graphs we can observe a slight decrease in PDR for higher node densities. This is due to the fact that we simply distribute the cumulative forwarding factor over all nodes within range by dividing by the number of neighbors. However, not all nodes necessarily have information to contribute. The higher the number of neighbors and therefore the more aggressive the scaling down of the forwarding factor, the more this effect comes into play. In future work we plan to do a mathematical analysis of this effect and incorporate it into the scaling of the forwarding factor.

IV. IMPACT OF TRANSMISSION RANGE

In the canonical configurations we have examined up to now we have assumed that each node broadcasts information to its closest neighbors, i.e., to two neighbors in the case of the circular network, and four neighbors in the case of the square grid network. Similarly, in the case of random networks, we assumed that the transmission range is relatively small compared to the size of the network. In this section we investigate how this assumption affects our results. In particular, we assume that all nodes transmit at an identical range ρ (using omni-directional antennas) but that ρ might allow to reach more than the closest neighbors.

In a wireless environment, the transmitted power P_T decays with distance as $\frac{P_T}{\rho^\gamma}$ due to path loss, where typical values are $\gamma \geq 2$. Thus, if a receiver at a distance ρ can successfully receive a signal that has power above a threshold P_0, then the

[2]In real scenarios, it is extremely unlikely that v' overhears none of the packets that its neighbors received previously to obtain their information. Furthermore, v' may obtain the missing information through a neighbor that is not within v's transmission range. Also this case is not part of the analysis. Therefore, the analysis below is a worst case estimate that gives an upper bound on the probability of v' not being able to decode after kg transmissions.

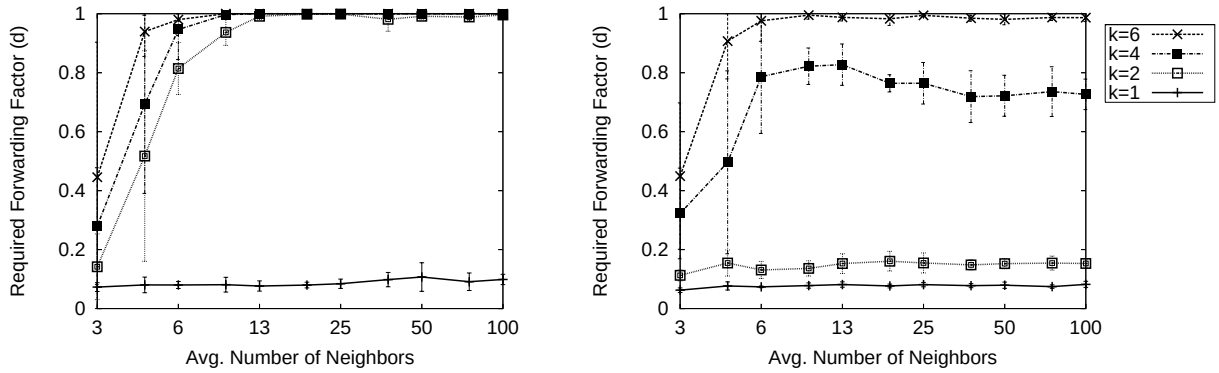

Fig. 5. PDR for different node densities with network coding (left) and probabilistic routing (right) for different forwarding factors (k) with *Algorithm 2A*

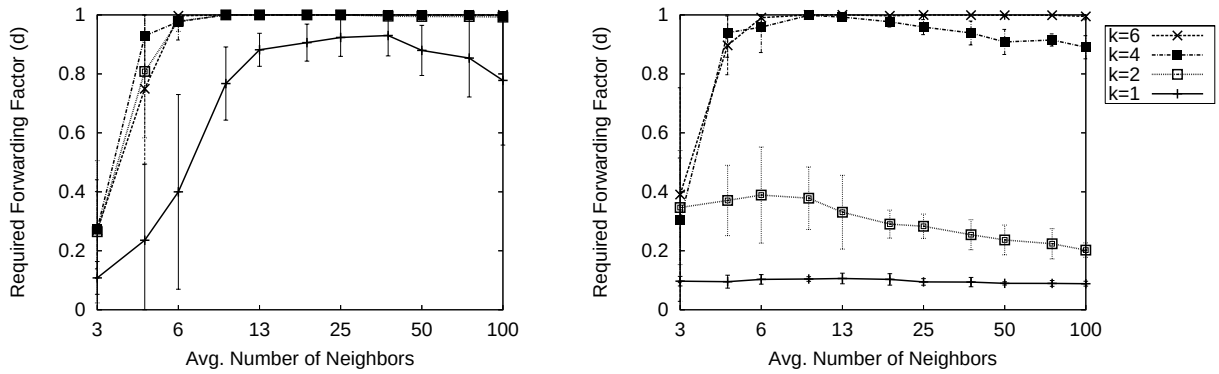

Fig. 6. PDR for different node densities with network coding (left) and probabilistic routing (right) for different forwarding factors (k) with *Algorithm 2B*

transmitted power P_T must increase proportionally to $P_0\rho^\gamma$. Increasing the range of transmission increases P_T. On the other hand, increasing the transmission range allows to reach more receivers during each transmission. In the following, we quantify this tradeoff.

A. Circular Network

In a circular network, to reach the two closest neighbors, a node needs to transmit at a radius of $2\sin(\frac{2\pi}{n}) = 2\sin(\theta)$. Generally to reach the $2k$ nearest neighbors, $1 \leq k \leq \frac{n}{2}$, a node needs to transmit at a radius of $2\sin(\frac{2\pi k}{n}) = 2\sin(k\theta)$. In the case of network coding, if each broadcast transmission reaches

- *the two closest neighbors*, we will need total power

$$P_2 = \frac{n-1}{2}\frac{P_0}{\sin^\gamma(\theta)},$$

- *the 2k closest neighbors*, we will need total power

$$P_{2k} = \frac{n-1}{2k}\frac{P_0}{\sin^\gamma(k\theta)}.$$

Thus,

$$\frac{P_2}{P_{2k}} = k(\frac{\sin(\theta)}{\sin(k\theta)})^\gamma = \frac{k}{k^\gamma}\frac{1-\frac{\theta^2}{3!}+\frac{\theta^4}{5!}-\ldots}{1-\frac{(k\theta)^2}{3!}+\frac{(k\theta)^4}{5!}-\ldots},$$

and for large n (small θ) we get that

$$\frac{P_2}{P_{2k}} \approx k^{1-\gamma}. \tag{8}$$

In the case of forwarding, if each broadcast transmission reaches $2k$ neighbors, we need in total power

$$P^f_{2k} = (1+\frac{n-1-2k}{k})\frac{P_0}{sin^\gamma(k\theta)}.$$

We conclude that in both cases we lose in terms of transmit power when increasing the transmission range, but the relative ratio $\frac{P^f_{2k}}{P_{2k}}$ remains equal to $\frac{1}{2}$, at least for k much smaller than n.

B. Square Grid

The square grid can be thought as 2 dimensional lattice $\mathcal{Z}^2$ (enveloping the surface of a torus) that contains all the points of the form $v = xe_1 + ye_2$, where x and y are integers and e_i are the vectors of the orthonormal basis, $e_1 = [1\ 0]$, $e_2 = [0\ 1]$. If we draw a circle in $\mathcal{R}^2$ with radius k around the point v it will contain all points (x, y) satisfying

$$(x-v_1)^2 + (y-v_2)^2 \leq k^2.$$

Thus, if we broadcast at a constant radius $\rho = k \in \mathcal{Z}$,the number of neighbors we can reach equals

$$N_k = \sum_{y=-k}^{y=k}(2\lfloor\sqrt{k^2-y^2}\rfloor+1)-1. \tag{9}$$

If we compare the number of transmissions that we need with and without network coding, we get that

$$\frac{T_{nc}}{T_w} = 1 - \frac{\sum_{y=0}^{y=k-1}(2\lfloor \min\{\sqrt{k^2-y^2}, \sqrt{k^2-(y-k)^2}\}\rfloor + 1}{\sum_{y=-k}^{y=k}(2\lfloor\sqrt{k^2-y^2}\rfloor + 1) - 1}$$

Values of this ratio are included in Table II.

TABLE II
CONVERGENCE OF RATIO $\frac{T_{nc}}{T_w}$.

k	1	2	10	50
$\frac{T_{nc}}{T_w}$	0.7500	0.6667	0.6013	0.6089

In the case of network coding, if each broadcast transmission reaches

- *the four closest neighbors*, we will need total power

$$P_1 = \frac{n-1}{4} P_0.$$

- *the k closest neighbors*, we will need total power

$$P_k = \frac{n-1}{N_k} \frac{P_0}{k^\gamma}.$$

Thus,

$$\frac{P_1}{P_k} = \frac{N_k}{4k^\gamma}. \tag{10}$$

If $\gamma \geq 2$ and using Eq. (9) we can see that $\frac{P_1}{P_k} \leq 1$.

We conclude that for $\gamma = 2$ increasing the transmission range does not affect the energy efficiency. For $\gamma > 2$ the optimal strategy in terms of power efficiency is to transmit to the closest neighbor. Moreover, as the transmission range increases ρ, the benefits network coding offers also increase and converge to approx. 0.609. This number is to the area that two circles of the same radius and centers at distance equal to the radius, intersect.

C. Random Network

In Figure 7 we show simulation results for a random network with 144 nodes and a fixed area of 1500m $\times$ 1500m. For each transmission range, we choose the smallest cumulative forwarding factor k for Algorithm 2B that results in an overall PDR of more than 99%. As can be seen from the left graph, with network coding higher transmission ranges even allow to *decrease* the total energy expenditure (assuming a path loss exponent of $\gamma = 2$). Recall that Algorithm 2B is only a heuristic and requires k to be somewhat larger than the optimal value. The intuition behind this result is that, the larger the transmit range, the more "regular" the network becomes in terms of number of neighbors, and the closer k can be set to the optimal value. Note that nodes can trade off the number of transmissions for transmit power, which in turn might allow for simpler MAC layer schedules.

In contrast, for flooding the overall energy consumption increases with the transmit range, since flooding does not allow to reduce the number of transmissions as aggressively as network coding for an increased number of neighbors.

The transmission range might also have an effect on delay. In the right graph of Figure 7 we see that there is a slight decrease in average (decoding) delay for flooding as well as for network coding, when the transmit range increases. This is the result of two factors: increasing the transmission range implies that more nodes can be reached by a single transmission. On the other hand, scheduling becomes more challenging, as the number of non-overlapping circles that can be simultaneously packed (i.e., transmissions during the same timeslot) is reduced.

V. IMPACT OF MOBILITY

In this section, we investigate what is the expected effect of mobility on the energy efficiency.

A. Theoretical Analysis

Consider the circular network in Fig. 1, where each broadcast transmission can bring new information to at most two receivers. We saw that, when forwarding, each broadcast transmission can bring new information to one new receiver, i.e., we cannot avoid having some overlap. We expect that if nodes are mobile, then as nodes move and their neighbors change, the amount of overlap will be reduced.

Mobility Model: We divide time into iterations. At the beginning of each iteration, we assume that nodes exchange positions on the circumference of the circle according to a uniform random permutation. That is, there exits n equidistant positions for nodes on the circle. Node i, after each permutation will go at any of the possible positions with uniform probability, and independently of what positions it has visited in the past.

The transmission policy is that during each iteration each node transmits once. We also assume that nodes do not know the information that their neighbors already have. Thus, without loss of generality, we can assume that during each iteration, and at each (possibly new) position, node i always broadcasts x_i.

We know that with network coding, to transmit the information from all sources to all receivers, it is sufficient to have each node transmit $\lceil \frac{n-1}{2} \rceil$ times (in the following we omit for simplicity the ceiling operation). We here try to investigate how mobility helps in the case of forwarding. In particular we will answer the following question. We consider a circular network with the described mobility pattern and where nodes employ forwarding. We assume that we allow a total of $m \geq \frac{n-1}{2}$ transmissions to each node. We are interested in the average number of receivers we expect each x_i to reach.

Without loss of generality consider node $i = 1$. We can think of the random permutation as selecting at each iteration two neighbors from the set $\{2, 3, \ldots n\}$. Thus, we can equivalently think of our problem as a "balls in bins" problem. It is easy to see that the probability node j is not a neighbor of node 1 during a given iteration is

$$1 - \frac{\binom{n-2}{1}}{\binom{n-1}{2}} = 1 - \frac{2}{n-1}. \tag{11}$$

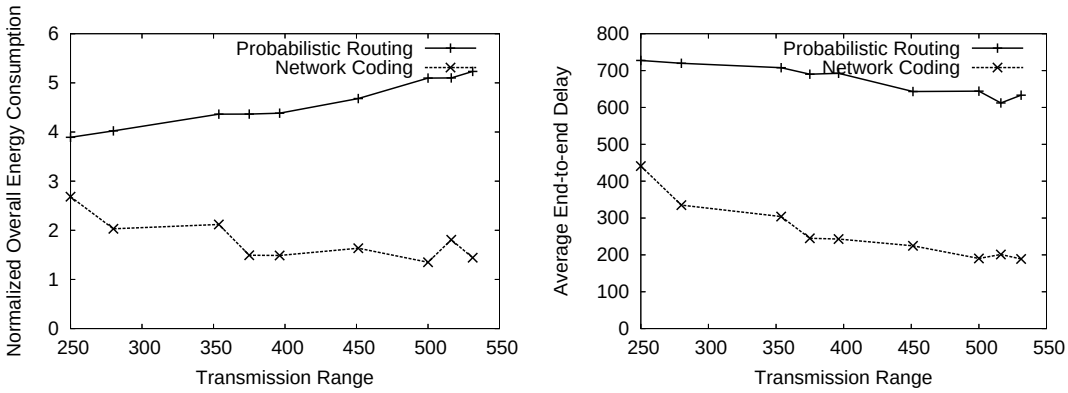

Fig. 7. Forwarding overhead in terms of energy consumption (i.e., number of transmissions $\times$ required transmit power) and decoding delay for different transmit ranges

Thus, the probability that x_1 is not received by node j after m iterations equals

$$(1 - \frac{2}{n-1})^m \approx e^{-\frac{2m}{n-1}}. \quad (12)$$

From symmetry, we get that the expected number of nodes that have not received x_1 equals $(n-1)e^{-\frac{2m}{n-1}}$.

Note that even for $m >> n$, we can see that there is always a non-zero probability that not all nodes will receive a certain symbol. At a first look this might seem surprising, as, when there is no mobility, n broadcast transmissions are sufficient to bring symbol x_1 to all receivers using forwarding. This "discrepancy" is due to the assumption that *nodes do not know what information their neighbors have*. This point is not crucial in the case of a static network, but makes a significant difference in the case of a mobile network.

In this argument we also assumed that during mobility nodes can only go in specific positions, and thus the number of neighbors of each node remains constant (equal to two) at each iteration. We next argue that this assumption is not restrictive.

Assume that each node can successfully broadcast over a range of $2\frac{2\pi r}{N}$ on the circumference of a circle with radius r. Also assume that at each iteration, nodes are placed uniformly at random on the circumference of the circle. Consider node 1. The probability that, at a particular iteration, node j is not placed within the broadcast radius of node 1 (i.e., node j cannot receive x_1) equals

$$1 - \frac{2\frac{2\pi r}{N}}{2\pi r} = 1 - \frac{2}{N}, \quad (13)$$

which coincides with Eq. (11). In other words, what changes in this case, is that the number of neighbors of a node is not exactly two, but on the average two. In the case of the square grid the same arguments apply,

B. Simulations over a Random Network

We analyze the impact of mobility through simulation using the random waypoint mobility model with a minimum speed of 2 m/s and a maximum speed of 10 m/s. We use network coding with $k = 3$ and probabilistic routing with $k = 6$ to achieve PDR's close to 100%. Figure 8 shows how the total number of received innovative packets evolves over time. As predicted, this number initially increases much quicker for both protocols when there is mobility. In contrast, towards the end of the simulation the rate of increase in a mobile network falls below that of a static network. For network coding with mobility, the last innovative packet is delivered at time 839, while it only takes 821 time steps in a static network. The corresponding numbers for probabilistic routing are 1586 and 1478, respectively.

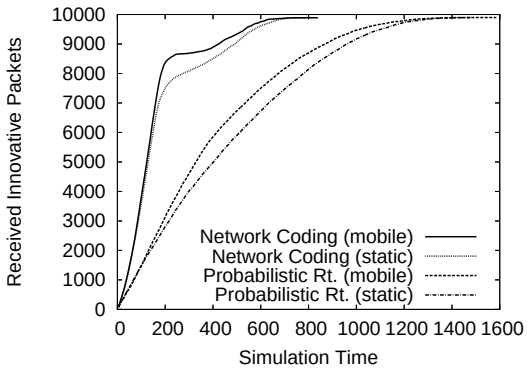

Fig. 8. Total number of received innovative packets over time for probabilistic routing and network coding for a static network and for a mobile network

VI. Practical considerations

A. Generation Management

Up to now we have assumed that each node is a source that has a single symbol to transmit, and that nodes are able to decode as soon as they receive n linearly independent combinations. Thus, all sources are decoded together at the end of the transmission.

In practice, the node memory and processing capabilities might be limited and therefore it might not be possible to combine all existing information symbols x_i in a single matrix. Moreover, in a random environment, there are perhaps benefits in combining symbols not only across space, but also *across time*, as is usually done in the network coding literature. Following the terminology in [14], it was shown in [13] that

Algorithm 2 can be easily extended to operate over generations. We define a generation as a collection of packets that we allow to be linearly combined. For example, we can think of our results up to now in the paper, as having generation that contains a single symbol from each source. In the other extreme, a generation might contain packets originating from a single source over time, i.e., we do not allow packets from different sources to combine. In general, each generation will contain a subset of packets, of a size that is determined by the memory and processing capabilities of the network nodes. In [18] the authors investigated and compared through simulation results several generation management methods. Our contribution in this paper is that we propose and evaluate distributed schemes to manage generations.

Generation Size Threshold

Without central control in the network, nodes have to manage generations based only on the information they already have. A node is responsible for choosing the right generation for each packet that originates at this node. To this end, the node checks which of the generations it knows of that have a size that does not exceed a certain threshold t. From these, it randomly picks one generation and inserts the packet into the corresponding matrix. If no such generation exists, the node creates a new generation with a random generation ID and inserts the packet.[3] The space of generation IDs has to be large enough so that the probability of having generations with the same ID created by different nodes is relatively small. (Note that does not prevent decoding of the two generations but merely "merges" them, leading to a larger generation size.)

The actual size of generations merely depends on the threshold t but is not limited by it. Several distant nodes may decide to insert packets into the same generation at the same time. Therefore, t needs to be adapted based on the average size of the matrices at a given node (and can be different for each node). Equivalently, t can be adapted based on the available memory at a node. The higher the probability of nodes inserting many new packets at the same time and the lower the node memory, the lower t needs to be.

To analyze this effect, we perform simulations on an area of 2000m $\times$ 2000m and with different numbers of nodes to obtain the different node densities. We use Algorithm 2B with $k = 3$. The left graph of Figure 9 shows the actual average size of generations at the nodes for different generation thresholds t. Particularly for small generations, the actual size exceeds t by a factor of 2-3. The ratio between t and the actual size of the generations is relatively independent of the network density. Only when the generation size is close to the number of nodes and the network is very dense, many generations may be created at the same time, which then contain fewer information vectors. This explains the drop in generation size for larger node densities.

[3]To avoid creating too many generations when many packets originate at different nodes at the same time, on can additionally impose a random delay a node has to wait before being allowed to create a new generation.

Local Generations

With local generations, nodes that may insert packets into a generation are limited to the λ-hop neighborhood of the node where the generation originated. Thus, we avoid having many nodes in different parts of the network insert packets into a generation at the same time.

A generation is created by a certain node v. The distance to this node determines generation membership. A hop count h_v is associated with the local decoding matrix for this generation and is initialized to zero. Vectors created from this matrix have a hop count of h_v+1. The hop count of the matrix at any node $v' \neq v$ is the minimum of the vectors' hop counts received for this generation, which corresponds to the minimum number of hops to reach the originator. The hop count can be transported in the packet header together with the encoding vector.

As before, a node determines if it is possible to include its information vector in an existing generation before starting a new generation. For this, it checks if it has a matrix with a hop count $h_v \leq \lambda$ and with a current size smaller than the threshold t. If no such matrix exists, the node starts a new generation. Nodes that are at a distance greater than λ hops might still form linear combinations (network code) packets from that generation but will not insert new packets into the generation. The impact of local generations on the average generation size is shown in the right graph of Figure 9.

In addition to the stricter limit on generation size, local generations also provide a slight improvement in PDR for very small generation size as shown in Figure 10. Local generations become more important when packet loss and communication patterns make communication with far away nodes in the network difficult or unlikely.

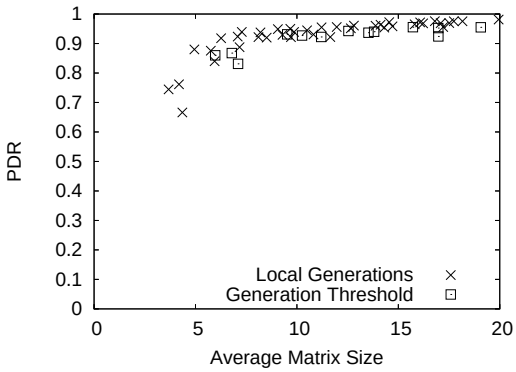

Fig. 10. PDR vs. average generation size with a generation size threshold and with local generations

B. Reducing Decoding Complexity

As discussed in Section II, to decode a "generation" of size g, i.e., g linearly independent equations, we need complexity $O(g^3)$, as we need to perform Gaussian elimination over the $g \times g$ matrix of the received coding vectors. If at each intermediate node we perform uniform at random combinations over $\mathcal{F}_q$, then the resulting matrix will be a random matrix with each element chosen uniformly at random over $\mathcal{F}_q$,

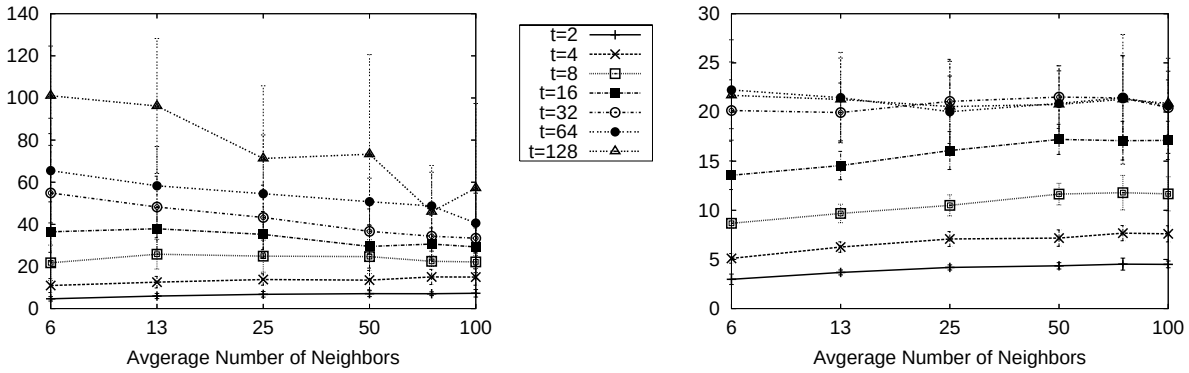

Fig. 9. Average generation size for different generation size threshold t and for different node densities (left graph) and with an additional hop count limitation (right graph)

In [19] it was observed that instead of choosing coding vectors uniformly over $\mathcal{F}_q$, in many cases we get comparable performance by performing sparse linear combinations, over a small field. This work was motivated by the observation [20] that a sparse random matrix of size $g \times g(1+\epsilon)$ with $lim_{g\to\infty}\frac{\epsilon}{g} = 0$, has with high probability full rank. In particular, this is true if we choose each element of the matrix independently to be one with probability $p = \frac{log(g)}{g}$, and zero otherwise. Moreover, such a matrix requires $O(g^2 log(g))$ operations to be decoded. If each node in the graph performs "sparse" linear combinations, we can express the resulting matrix that a receiver needs to decode as a product of sparse matrices which we can solve sequentially. Here we examine the effect of reducing the alphabet size and of forming "sparse" linear combinations through simulation results.

Reducing the Alphabet Size: From simulations with 100 nodes, a generation size of 100, and on average 12 neighbors per node, we see that a relatively small alphabet size is sufficient to achieve good network coding performance. Only the field of size two, which is much smaller than the average number of neighbors, provides an insufficient number of linearly independent combinations per neighborhood. Already an alphabet size of 2^2 comes close to the performance of an alphabet size of 2^8 which is what we used in all of the previous simulations.

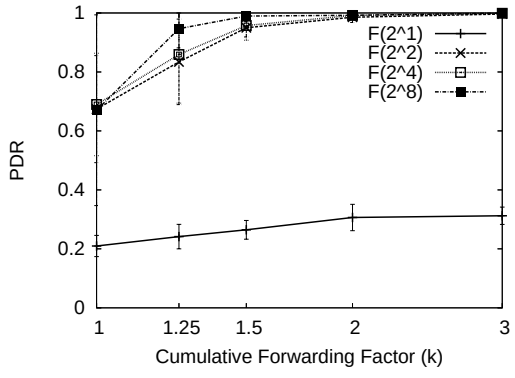

Fig. 11. Impact of reducing the alphabet size on PDR

Reducing the Matrix Density: We use the following algorithm to generate vectors with a limited number of non-zero entries. As long as the number of non-zero coefficients is lower than a threshold q, a row is randomly picked from the decoding matrix, multiplied by a random coefficient, and added to the vector to be sent out. We use a simulation setting similar to that of the previous paragraph. Setting $q = 1$ corresponds to sending out the information of a single row of the decoding matrix which is non-innovative for neighboring nodes with a high probability (in fact, performance degrades to that of probabilistic routing). As soon as $q \approx log(g)$, there is little difference in performance compared to an unrestricted generation of vectors ($q = 100$).

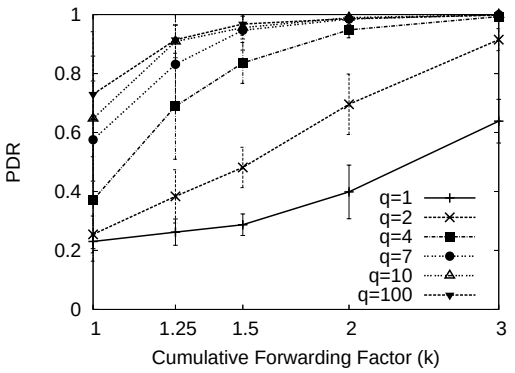

Fig. 12. Impact of reducing the matrix density on PDR

VII. Conclusions

In this paper we characterized the minimum amount of energy required to transmit one unit of information from a source to all receivers for canonical configurations, and developed distributed algorithms that allow to approach the optimal performance in practice. The emphasis of the paper was in tradeoffs and design choices that arise in practical systems, such distributed generation management, effect of transmission range and mobility. Our work indicates that there is a potential for significant benefits, when deploying network coding over a practical wireless ad hoc network environment.

REFERENCES

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, July 2000.

[2] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, Feb. 2003.

[3] C. Diot, J. Scott, E. Upton, and M. Liberatore, "The Haggle architecture," Intel Research Cambridge, Tech. Rep. IRC-TR-04-016, 2004.

[4] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *ACM/IEEE Mobicom*, Aug. 1999.

[5] M. Cagalj, J.-P. Hubaux, and C. Enz, "Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues," in *ACM/IEEE Mobicom*, 2002.

[6] Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," in *IEEE WCNC*, Mar. 2003.

[7] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," in *IEEE Infocom*, June 2002.

[8] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *ACM Wireless Networks Journal*, Sept. 2002.

[9] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, Jan. 2002.

[10] J. Wu and F. Dai, "A generic distributed broadcast scheme in ad hoc wireless networks," *IEEE Transactions on Computers*, Oct. 2004.

[11] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," in *IEEE Information Theory Workshop*, Oct. 2004.

[12] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," in *IEEE Infocom*, Mar. 2005.

[13] J. Widmer, C. Fragouli, and J. Y. Le Boudec, "Energy efficient broadcasting in wireless ad-hoc networks using network coding," *First Workshop on Network Coding*, 2005.

[14] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Allerton*, Oct. 2003.

[15] C. H. Lim and P. J. Lee, "More flexible exponentiation with precomputation," in *Proc. Advances in Cryptology: 14th Annual International Cryptology Conference*, Aug. 1994.

[16] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," in *IEEE Infocom*, June 2002.

[17] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *ISIT*, June 2003.

[18] J. Widmer and J.-Y. L. Boudec, "Network coding for efficient communication in extreme networks," in *Workshop on delay tolerant networking and related networks (WDTN-05)*, Philadelphia, PA, Aug. 2005.

[19] C. F. Payam Pakzad and A. Shokrollahi, "Coding schemes for line networks," *ISIT*, 2005.

[20] R. K. J. Blömer and E. Welzl, "The rank of sparse random matrices over finite fields," *Random Structures Algorithms 10*, 1997.

APPENDIX: COMPUTATION OF COVERAGE PROBABILITY

Let D_0 be a diskof radius 1. Let $j = kg$, and $D_1, ..., D_j$ be j disks, also of radius 1, with centers in D_0, drawn independently and uniformly in D_0. Define Q_j^g as the probability that a random point M in D_0 is covered by less than g of the j disks. We are interested in Q_{kg}^g. We have:

$$Q_j^g = \sum_{i=0}^{g-1} q_j^i \tag{14}$$

with q_j^i equal to the probability that a random point in D_0 is covered by exactly i of the j disks. Further:

$$q_j^i = \int_{D_0} \mathbb{P}\left(m \text{ is covered by exactly } i \text{ of the } j \text{ disks}\right) \frac{dm}{\pi} \tag{15}$$

By independence of D_1,..., D_j:

$$q_j^i = \int_{D_0} \left(\begin{array}{c} j \\ i \end{array} \right) \left(1 - \mathbb{P}(m \notin D_1)\right)^i \left(\mathbb{P}(m \notin D_1)\right)^{j-i} \frac{dm}{\pi} \tag{16}$$

We now compute $\mathbb{P}(m \in / D_1)$. By circular symmetry, it depends only on the distance $\rho = \|m\|$ from 0 to m. Let $p_1(\rho)$ be the value of $\mathbb{P}(m \notin D_1)$ when $\|m\| = \rho$. To compute $p_1(\rho)$ we first compute $p(\rho, r)$, defined as the probability that $m \notin D_1$ given that the distance from the center of D_1 to the origin is r. This is obtained by considering a random experiment where we select the center ω_1 of D_1 uniformly on the circle centered at 0 with radius r. Let θ be the principal measure of the angle from $\vec{O\omega_1}$ to $\vec{Om}$. Let $\theta_{\max}$ be the maximum value of θ such that $m \in D_1$. We have

$$\left\{ \begin{array}{l} \text{if } r + \rho > 1 \text{ then } \theta_{\max} = \arccos \frac{r^2+\rho^2-1}{2r\rho} \\ \text{else } \theta_{\max} = \pi \end{array} \right. \tag{17}$$

and

$$p(\rho, r) = 1 - \frac{2\theta_{\max}}{2\pi} = 1 - \frac{\theta_{\max}}{\pi} \tag{18}$$

Thus (taking into account that r is a polar coordinate):

$$\begin{array}{rl} p_1(\rho) & = 2\int_0^1 p(\rho, r) r dr \\ & = 2\int_{1-\rho}^1 r \left(1 - \frac{1}{\pi} \arccos \frac{r^2+\rho^2-1}{2r\rho}\right) dr \\ & = \frac{\rho\sqrt{4-\rho^2} + 4 \arcsin \frac{\rho}{2}}{2\pi} \end{array} \tag{19}$$

and

$$q_j^i = 2 \left(\begin{array}{c} j \\ i \end{array} \right) \int_0^1 \left(1 - p_1(\rho)\right)^i \left(p_1(\rho)\right)^{j-i} \rho d\rho \tag{20}$$

The integral in Equation (20) can be computed in closed form for every fixed value of i and j. Note that $Q_j^1 = q_j^0$ is the probability of no coverage by j disks, which was computed exactly for $j = 1$ and by simulation for larger j in [4]. In contrast, we obtain exact closed forms for all values of i and j. For example the first values of $Q_j^1 = q_j^0$ are:

$$\begin{array}{rcl} q_1^0 & = & \frac{3\sqrt{3}}{4\pi} \\ q_2^0 & = & \frac{-5+3\sqrt{3}\pi}{6\pi^2} \\ q_3^0 & = & \frac{-351\sqrt{3}+152\pi+24\sqrt{3}\pi^2}{96\pi^3} \\ q_4^0 & = & \frac{21343-7020\sqrt{3}\pi+1520\pi^2+160\sqrt{3}\pi^3}{1440\pi^4} \\ q_5^0 & = & \frac{4004397\sqrt{3}-1363380\pi-210600\sqrt{3}\pi^2+30400\pi^3+2400\sqrt{3}\pi^4}{51840\pi^5} \end{array}$$

We can also compute limits for large g or large k. We have $Q_j^g = 2\int_0^1 B(kg, g, 1 - p_1(\rho))\rho d\rho$ where $B(j, g, p)$ is the (binomial) probability that a random experiment with success proability p succeeds less than g times in j experiments. For large k or g, we can approximate $B(kg, g, p)$ by a normal distribution, which gives ($Q(x)$ is the probability that a standard normal random variable is larger than x)

$$\begin{array}{rl} Q_{kg}^g & \approx 2\int_0^1 (1 - Q\left(\sqrt{g}\frac{1 - k(1 - p_1(\rho))}{\sqrt{k(1 - p_1(\rho))p_1(\rho)}}\right) \rho d\rho \\ & \approx 2\int_0^1 Q\left(\sqrt{gk}\sqrt{\frac{1}{p_1(\rho)} - 1}\right) \rho d\rho \end{array} \tag{21}$$

$$= \frac{2}{\sqrt{2\pi}} \int_0^1 \int_{\sqrt{gk}\sqrt{\frac{1}{p_1(\rho)}-1}}^{\infty} e^{-\frac{s^2}{2}} ds d\rho \tag{22}$$

By inversion of the order of integration, one obtains

$$Q_{kg}^g \approx 2 \int_{\sqrt{gk\gamma}}^{\infty} e^{-\frac{s^2}{2}} (1 - \phi(s)^2) ds \tag{23}$$

where $\phi(s)$ is implicitly defined by $p_1(\phi(s)) = \frac{1}{\sqrt{1+s^2/gk}}$ and $\gamma = \sqrt{\frac{1}{p_1(1)} - 1} \approx 0.642042$. This function increases from 0 (for $s = s_0 = \sqrt{gk\gamma}$) to 1 for $s \to \infty$; we approximate it with the piecewise linear function given by the derivative at s_0 and the asymptote which corresponds to $\phi(s)$ close to 1. One obtains the approximation $1 - \phi(s)^2 \approx (s - s_0)\frac{\alpha}{\sqrt{gk}}$ for $s_0 \leq s \leq \frac{\sqrt{gk}}{\alpha} + s_0$ with

$$\alpha = \frac{\left(\sqrt{3} + \frac{2\pi}{3}\right)^2 \sqrt{\frac{1}{3}\left(-1 + \frac{6\pi}{3\sqrt{3}+2\pi}\right)}}{\pi} \approx 2.15607$$

and otherwise $1 - \phi(s)^2 \approx 1$. The resulting integral can be computed and one finds

$$Q_{kg}^g \approx \frac{2\alpha}{\sqrt{2\pi gk}} e^{-\frac{1}{2}gk\gamma} = \frac{1.72029}{\sqrt{gk}} e^{-0.321021gk} \tag{24}$$

With a similar analysis, for large g, we have the following limits, when k is fixed:

- For $k = 1$: $\lim_{g\to\infty} Q_g^g = 1$
- For $k = 2$, $\lim_{g\to\infty} Q_{2g}^g = 1 - \rho_1^2 \approx 0.347224$ where ρ_1 is the root in $(0, 1)$ of the equation $p_1(\rho_1) = 1/2$.
- For $k \geq 3$: $\lim_{g\to\infty} Q_g^g = 0$