Low-complexity energy-efficient broadcasting in wireless ad-hoc networks using network coding

Jörg Widmer, Christina Fragouli, and Jean-Yves Le Boudec
Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland

***Abstract*— Energy efficiency, i.e., the amount of battery energy consumed to transmit bits across a wireless link, is a critical design parameter for wireless ad-hoc networks. We examine the problem of broadcasting information to all nodes in an ad-hoc network, when a large percentage of the nodes act as sources. We theoretically quantify the energy savings that network coding can offer for the cases of two regular topologies. We then propose low-complexity distributed algorithms, and demonstrate through simulation that for random networks, network coding can in fact offer significant benefits in terms of energy consumption.**

I. Introduction

Network coding was recently proposed in [1]. With network coding, intermediate nodes in the network not only forward but also process the incoming information flows, which may result in significant benefits. Wireless ad-hoc and sensor networks are likely to be among the first areas where network coding will be applied, as these environments offer more freedom in terms of protocol design choices. The wireless environment introduces both new challenges and new capabilities. Challenges include interference from other users, power constraints, and fading channels. Possible advantages are the natural capability of broadcasting, making use of node mobility, etc. An important factor in reaping benefits from these depends on the development of low-complexity scalable algorithms that are suited to the wireless environment.

We propose simple distributed algorithms for broadcasting in wireless networks. Such one-to-all communication is essential as a discovery mechanism at the network or application layer [2]. For example, it is used for route discovery in on-demand ad-hoc routing protocols. We show that network coding can offer significant benefits in terms of energy efficiency, which is perhaps the most important design metric for devices in wireless ad-hoc networks.

The problem of minimum energy broadcasting in ad-hoc wireless networks is NP-complete [3] and a large number of approximation algorithms have been proposed. Usually, these are either based on probabilistic algorithms where packets are only forwarded with a certain probability [4], [5], [6], or some form of topology control to form connected dominating sets of forwarding nodes [7], [8]. Flooding, which is the simplest possible algorithm, in wireless networks results in a prohibitively large overhead [4].

If we allow intermediate nodes to code (i.e., we use network coding), the problem can be formulated as a linear program and thus accepts a polynomial-time solution. The authors in [9] present a linear program formulation for the problem of minimizing the energy per bit when multicasting in an ad-hoc wireless network. An alternative formulation for minimum-energy multicast in wireless networks is presented in [10], where a distributed algorithm to select the minimum-energy multicast tree is proposed.

Our focus is on broadcasting, which is a special case of multicasting. Our work can also be thought of as an application of [11] and [12] in the case of wireless ad-hoc networks. We propose low complexity distributed algorithms and evaluate their performance in terms of energy efficiency. To motivate the proposed algorithms we start by examining in Section II structured configurations, such as the rectangular grid network. We describe our algorithms for the general case in Section III and present simulation results in Section IV.

II. Canonical Examples

This section presents two examples, the circular network and the rectangular grid network, that motivate our algorithms and demonstrate the potential of network coding. In both cases we consider a wireless ad-hoc network with n nodes. We assume that every node is a source, that wants to broadcast information to all other nodes. Also, each node broadcasts at a fixed range and therefore, each transmission consumes the same amount of energy.

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 $\frac{T_{nc}}{T_w}$.

A. Circular Network

Consider n nodes placed at equal distances around a circle as depicted in Fig. 1. Assume that each node can successfully broadcast information to its two neighbors. For example, node b_1 can broadcast to nodes a_1 and a_2.

Lemma 1: For the circular network it holds that

1) without network coding $T_w \geq n-2$
2) with network coding $T_{nc} \geq \frac{n-1}{2}$.

Proof: Since a node can successfully broadcast to its two nearest neighbors, each broadcast transmission can transfer at most one unit of information to two receivers. We have $n-1$ receivers to cover and thus the best energy efficiency we may hope for is $\frac{n-1}{2}$ per information unit.

When forwarding w.l.g. we may consider a single source broadcasting to $n-1$ receivers. The first transmission reaches

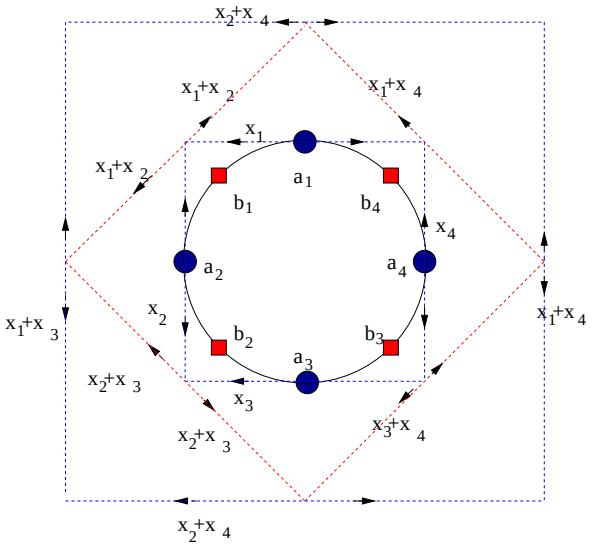

Fig. 1. A circular configuration.

two receivers. Each additional transmission can contribute one unit of information to one receiver. ■

For the case of forwarding, it is easy to see that a simple flooding algorithm achieves the bound in Lemma 1.

For network coding consider the following scheme. Assume that n is an even number. Partition the n nodes in two sets $A = \{a_1, \dots a_{\frac{n}{2}}\}$ and $B = \{b_1, \dots b_{\frac{n}{2}}\}$ of size $\frac{n}{2}$ each, such that every node in A has as nearest neighbors two nodes in B. For example, Fig. 1 depicts a circular configuration with $n = 8$ nodes. It is sufficient to show that we can broadcast one information unit from each node in set A to all nodes in sets A and B using $T_{nc} \geq \frac{n}{2}$ transmissions. We can then repeat this procedure symmetrically to broadcast the information from the nodes in B.

Let $\{x_1, \dots, x_{\frac{n}{2}}\}$ denote the information units associated with the nodes in A. Consider the following transmission scheme that operates in $\frac{n}{4}$ steps. Each step has two phases, where first nodes in A transmit and nodes in B receive and then nodes in B transmit and nodes in A receive. For simplicity of notation, we assume that all indices are $\mod \frac{n}{2}$.

Algorithm NC1

Step 1*:*

- Phase 1: The nodes in set A transmit their information to their nearest neighbors, such that each node b_i receives x_i and x_{i+1} as shown in Fig. 1.
- Phase 2: The nodes in set B simply add the information symbols they receive and broadcast it. For example, node a_i receives $x_{i+1} + x_i$ from node b_i and $x_{i-1} + x_i$ from node b_{i-1}. Thus, node a_i has the information units from sources a_{i-1} and a_{i+1}.

Step k, $k > 1$*:*

- Phase 1: Each node in A transmits the sum of the two information units it received in phase 2, step $k-1$.
- Phase 2: Each node in B transmits the sum of the two information units it received in phase 1, step k.

Lemma 2: There exist schemes that achieve the lower bounds in Lemma 1. Thus $\lim_{n\to\infty} \frac{T_{nc}}{T_w} = \frac{1}{2}$.

Proof: Given the previous discussion it is sufficient to show that Algorithm $NC1$ achieves the bound in Lemma 1. It is easy to see that Algorithm $NC1$ will conclude in at most $\frac{n}{4}$ steps. Each step involves n transmissions and each transmission brings one new information unit to two receivers. Note that in the last step, second phase, fewer transmissions are required, but this will not affect the order of the result. This scheme transmits $\frac{n}{2}$ information units in $n\frac{n}{4}$ transmissions, and thus, $T_{nc} = \frac{n}{2}$. ■

Assume now that each node can transmit at a constant radius of k neighbors, i.e., each transmission reaches at most $2k$ nodes. Consider set A to consist of $\frac{n}{2k}$ sources that are $2k$ nodes apart, and set B to consist of $\frac{n}{2k}$ "relays". Each node in B is at distance k from its two closest neighbors in set A. We then effectively reduce our problem to the $k = 1$ case and we get directly the following theorem.

Theorem 1: In a circular network where each node can broadcast information at a constant radius of k neighbors

$$\frac{T_{nc}}{T_w} = \frac{\frac{n-1}{2k}}{\frac{n-1}{k} - 1} \approx \frac{1}{2}. \quad (1)$$

Note that adjusting the transmit power to reach further than the direct neighbors is not energy efficient. It allows to reduce the number of transmissions by a factor of k, but the received power decays proportional to k^l with a path loss exponent $l \approx 2$ (depending on the environment). In total, such a scheme requires a factor of k^{l-1} more energy.

B. Rectangular Grid

Consider a wireless ad-hoc network with n nodes. Let $n = m^2$. Assume that every node is a source, that wants to broadcast information to all n nodes. The nodes are placed on the vertices of a rectangular grid and each node can successfully broadcast information to its four nearest neighbors. We are interested in a transmission scheme that minimizes the total expended energy.

Lemma 3: For the rectangular grid network it holds that:

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

Proof: Each broadcast transmission can contribute one unit of information to at most four receivers. However, when forwarding we have an overlap of at least one receiver, i.e., each broadcast transmission can contribute one unit of information to at most three receivers. ■

Lemma 4: There exist schemes that achieve the lower bounds in Lemma 3 and thus $\lim_{n\to\infty} \frac{T_{nc}}{T_w} = \frac{3}{4}$.

Proof: For the case of forwarding, we can simply use flooding along one horizontal line (direction), and along perpendicular lines.

For the case of network coding, we extend the proof idea in Lemma 2. We partition the square lattice into sub-lattices A and B, such that the four closest neighbors for an element in A belong to B (and vice-versa). Let nodes A be sources. We will describe a scheme that transmits one information unit from all

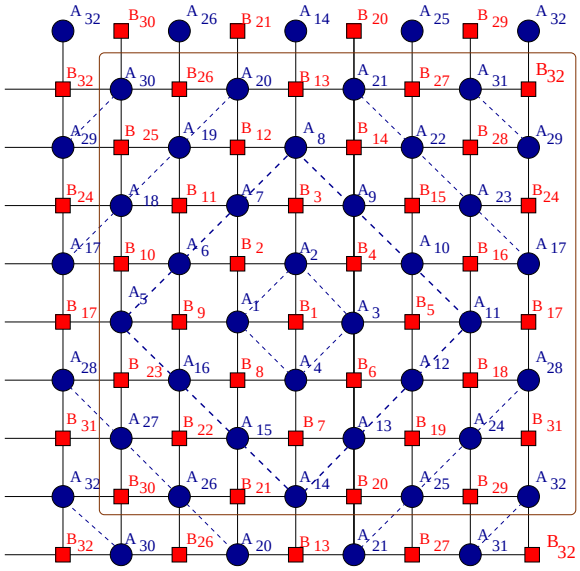

Fig. 2. A rectangular grid configuration.

sources in A to all nodes in A and B. Again consider steps divided in two phases, where in the first phase the nodes in A transmit, while in the second phase, the nodes in B transmit. To avoid edge effects, assume that the square grid envelopes the surface of a torus. Fig. 2 depicts an example network with 64 nodes enveloping the surface of a torus. For example, the closest neighbors of the node B_{32} are the nodes A_{29}, A_{30}, A_{31} and A_{32}.

We now make the connection with the circular network proof. By "distance" between two nodes we refer to the number of hops that separate them. For any node x_0 in the circular network, the number of neighbors at distance k from x_0 is two, independent of k (with a possible exception for $k = \frac{n}{2}$, i.e., when wrapping around, where we may have only one new neighbor). Thus, at step k in the proof of Lemma 2, it is sufficient for x_0 to receive two new information units, to learn the information of the neighbors at distance $2k$.

In contrast, in an infinite square lattice, the number of neighbors N_k at distance k is $N_k = N_{k-1} + 4$, $N_1 = 4$, $k \geq 2$ (called the coordination sequence of the square lattice.) In the case of a grid with m^2 points placed on the surface of a torus, the number of new neighbors increases up to a point, and then, because of overlap when wrapping around, starts decreasing. (We assume hereafter that m is even, but very similar arguments hold for m odd.) Similar to the circular network, our algorithm for the rectangular grid case consists of steps. In each step, every node collects information from sources that are at an increasing distance from it. However, since in this case, unlike the circular network, the number of neighbors depends on the distance, the number of transmissions per node at each step is not constant.

In the following, we first work out in detail the example in Fig. 2, and then formally present our algorithm. As can be seen from Table I, a node $b \in B$ has 4, 12, 12, and 4 (new) neighbor sources at distances 1, 3, 5 and 7 respectively. (Because the network is symmetric, these numbers are the same for all $a \in A$ and $b \in B$.) Fig. 2 depicts with dashed rectangles how during each step a node in B collects information from an increased distance.

Step 1:

- Each B_i receives the information from its four direct neighbors in A. For example, B_1 receives the information from sources A_1, A_2, A_3, A_4.
- Each B_i transmits two linear combinations of the four information units he receives. Each A_i receives 8 linear combinations, and solves a system of linear equations to retrieve the information from all sources at distance two. For example, A_2 has the information from the sources A_1, A_4, A_3, A_{10}, A_9, A_8, A_7, and A_6.

Step 2:

- Each A_i transmits three linear combinations from the 8 information units it previously received. As a result, each B_i is able to decode the information from the 12 sources at distance three. For example B_1 receives the information from the sources in the second larger dashed rectangular in Fig. 2.
- Each B_i transmits four linear combinations from the 12 sources it received. Each A_i receives 16 linear combinations and decodes 14 variables (here we transmit some redundant linear combination, but their number is negligible.)

Step 3:

- Each A_i transmits three linear combinations from the 14 it received. Each B_i decodes 12 information sources that are at distance five.
- Each B_i transmits two linear combinations. Each A_i receives 8 linear combinations and decodes the 8 sources at distance six.

Step 4:

- Each A_i transmits transmits one linear combination. Each B_i decodes the four sources at distance seven. Each B_i has now all the sources.
- The A_i's have all the sources but for one source. For example, A_2 does not have the information of source A_{32}. A subset of the B_i's transmits one linear combination that brings the missing information to all A_i.

The general algorithm can be described as follows. In the general case, the number of (new) sources at different steps k (and corresponding distances) for nodes in A and in B are described in Table I.

TABLE I

NUMBER OF NEW SOURCES AT NC2.

step k	**neighbors** N_{2k-1} for $a \in A$	**neighbors** N_{2k} for $b \in B$
$k = 1$	4	8
$1 < k < \frac{m}{4} - 1$	$N_{2k-2} + 4$	$N_{2k-1} + 4$
$k = \frac{m}{4}$	$N_{2k-2} + 4$	$N_{2k-1} + 2$
$k = \frac{m}{4} + 1$	$N_{2k-2} - 2$	$N_{2k-1} - 4$
$\frac{m}{4} + 1 < k < \frac{m}{2}$	$N_{2k-2} - 4$	$N_{2k-1} - 4$
$k = \frac{m}{2}$	4	1

Algorithm NC2

Step 1*:*

- Phase 1: The nodes in set A transmit their information to their four nearest neighbors. Each node in A transmits once. Each node in B receives four messages.
- Phase 2: Each node in set B broadcasts two linear combinations of the four new messages it has. Each node in A receives eight linear equations, that it can solve to retrieve the information from the eight distance-two neighbors.

Step k, $1 < k \leq \frac{m}{2}$*:*

- Phase 1: Each node in A transmits $\lceil \frac{N_{2k-1}}{4} \rceil$ linear combinations from the N_{2k-2} information units it received in phase 2, step $k-1$. Each node in B receives the information units from all sources at distance $2k-1$.
- Phase 2: Each node in B transmits $\lceil \frac{N_{2k}}{4} \rceil$ from the N_{2k-1} information units it received in phase 1, step k. Each node in A receives the information units from all sources at distance $2k$.

Thus, for $k = 1, \ldots \frac{m}{2}$, each node collects the information from a constantly increasing area.

Finally, the only remaining point to prove is, that there exist linear combinations such that each receiver is able to decode. With a centralized scheme, we are asking for vector coefficients such that a product of determinants is nonzero. We can address this problem using algebraic tools as for example in [13]. Indeed, it is clear from the min-cut max-flow theorem that no determinant is identically zero. Using Lemma 2 in [13] the result follows.

Alternatively, we can use randomized coding, where each node transmits random linear combinations of the information symbols he received in the previous phase. Each receiver at every step receives N linear combinations of N variables, and with high probability has a full rank system of equations. Note that with randomized coding, this scheme becomes decentralized: at each step, each node transmits linear combinations independently of what the neighbors transmit. Moreover, the number of transmissions at every step is a function exclusively of the step number. ■

In analogy to Theorem 1, Algorithm $NC2$ can be extended to a radio range that covers more than four neighbors. We are currently investigating coding schemes for this case.

III. Algorithms for General Networks

In the previous section we formally proved the optimality of two low-complexity network coding algorithms. In this section we present further algorithms that have even lower complexity, achieve the same performance in square grid networks, but are also well suited for random topologies.

Algorithm NC3

- Each node maintains a send counter s that determines the number of broadcasts a node is allowed to send. Each broadcast reduces s by 1 and a node is not eligible for sending if $s < 1$. These broadcasts contain linear combinations of all the information available at a node. Initially, $s = 0$.
- An information unit that originates at a node increases s by one. The information is broadcasted *uncoded* (i.e., not combined with any other information).
- For each innovative packet a node receives, s is incremented by d.

The parameter d determines the ratio of sent packets to received innovative packets. Algorithms *NC1* and *NC2* distribute information with the smallest possible d. In real wireless networks with irregular topologies and packet loss, it is helpful to use a larger d for additional robustness. For example, with a very low value for d, a node that has fewer than the necessary number of neighbors may also prevent other nodes from being able to decode by not sending out enough broadcasts himself.

Good values for d depend on the network topology and in particular on the node density. For example, a node that has a neighbor which has no other neighbors needs to forward each innovative packet to this neighbor. While determining the optimal value of d for a node requires global knowledge, it is easy to find values for d that work reasonably well for all nodes in the network, solely based on the average node density. The smaller the value of d, the lower the total number of broadcasts and the higher the probability, that nodes in sparsely populated areas of the network will not be able to decode all packets.

When no information about the average node density is available or nodes have to be able to decode with a high probability, it is necessary to choose a large value for d. In random networks, the major part of the protocol overhead then comes from non-innovative packets in dense parts of the network, where few broadcasts would suffice. Algorithm *NC4* reduces the number of such useless broadcasts.

Algorithm NC4

- In addition to the send counter as in Algorithm *NC3*, nodes use a receive counter. For each c non-innovative packets a node receives, the send counter s is decremented by one.

This algorithm effectively adapts d to the local node density. It limits the number of non-innovative broadcasts per neighborhood while sacrificing little reliability.

Since we are interested in low-complexity algorithms, we compare our protocols against probabilistic broadcasting schemes [4], [6]. We consider three algorithms.

Algorithm FL1

- An information unit that originates at a node is broadcast to the neighbors.
- A node rebroadcasts a new packet with probability d.

Algorithm FL2

- As Algorithm *FL1*, but when a node overhears c duplicate transmissions of the same packet, it removes the corresponding broadcast from the interface queue. (if it has not yet been sent).

These first two are in analogy to Algorithms *NC3* and *NC4*. For the last flooding algorithm, we assume that a node has additional information about the current state of its neighborhood (and that there is no cost associated with obtaining this information).

Algorithm FL3

- As Algorithm *FL1*, but when all the neighbors of a node have received a given information unit, the node removes the corresponding broadcast from the interface queue. (if it has not yet been sent).

A node thus avoids unnecessary broadcasts which reduces the overhead required for the same packet delivery ratio.

IV. Simulation Results

We have implemented the decentralized network coding and flooding algorithms using a network simulator. A packet transmission takes exactly one time unit. We assume that a node can either send or receive and it can only send or receive one packet at a time. Nodes have a nominal radio range of 250m. Transmissions are broadcasted and are received by all neighbors (i.e., all nodes within transmission range). 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. The simulation area has a size of 1500m$\times$1500m. The number of nodes is 144 which, for the regular grid, results in exactly 12 neighbors per node. Each node has one information unit to send to all nodes. For the network coding, we use a fixed field size of 2^{16} and transport the encoding vectors in the packet header as suggested in [14].

We ignore edge effects by placing the nodes on a torus. All the simulations were also performed without this assumption, which reduces the average node density within a node's neighborhood and increases the average path length. Apart from that, the findings of the simulations remain unchanged and we therefore omit these results.

A. From Grids to Random Networks

We first investigate the impact of the send count increment d on protocol performance in regular and random networks. We vary d from 0 to 1 and show the resulting average number of transmissions T and the packet delivery ratio in Fig. 3. Each point in the curves corresponds to a specific value of d. Note that $d(n-1)+1$ is an upper bound on the transmissions per node, if each node has a packet to send. The lower the packet delivery ratio, the further T will be from this upper limit.[1]

Note that with network coding, a node might not be able to decode all the innovative packets it receives. For the regular grid and $d < 1/12$, the algorithm stops very early due to the low number of packets, while the flooding algorithms already achieve delivery ratios of up to 14%. As soon as $d > 1/12$, the delivery ratio of network coding jumps to 100%, while flooding is still below 20%. To deliver all packets in the regular grid, $T_{NC3} = nd = 144/12$, which confirms the analysis of Section II. Flooding and idealized flooding achieve 100% delivery ratio for $T_{FL1} = T_{FL3} = 108$, although idealized flooding gets very close to 100% delivery ratio with significantly fewer transmissions than normal flooding.

Generally, a higher number of transmissions is required to achieve these packet delivery ratios in random topologies. Nevertheless, network coding again clearly outperforms both flooding algorithms. For intermediate packet delivery ratios, network coding incurs roughly half the overhead of flooding and $2/3$ of the overhead of idealized flooding. This gap widens for high delivery ratios. With flooding, almost every newly received packet is rebroadcasted, which requires close to 144 transmissions. The assumption of additional information for idealized flooding allows to achieve the same performance for $T_{FL3} = 50$ packets. Even without this information, network coding remains below 40 transmissions per node.

We also performed the same simulations using Algorithm *NC4* and *FL2* with a receive count limit of $c = 5$. The results (not shown here) show a slight improvement in the number of transmission of network coding in random topologies in the regime of high delivery ratios. Here, 100% delivery ratio is achieved for $T_{NC4} = 33$ instead of 40. Improvements are similar for flooding, the most significant being a 100% delivery ratio for $T_{FL2} = 94$ instead of 144. Since idealized flooding is in fact a more intelligent version of a receive count limit, its performance remains unchanged. Nevertheless, the large difference in performance between network coding and the flooding algorithms remains unchanged.

B. Impact of Packet Loss

Here we examine the impact of random packet loss often prevalent in wireless networks due to fading, interference, etc. We drop packets at receiving nodes with a fixed probability between 0 and 100%. We further use a send count increment of $d = 0.5$ and a receive count limit of $c = 5$.

The corresponding simulation results are shown in Fig. 4. Network coding is insensitive to packet loss rates of up to 40% and continues to achieve high delivery ratios for drop rates of

[1]Flooding requires $d = 1$ to achieve a packet delivery ratio of 1, independent of the network topology. Otherwise, there is a certain (small) probability, that all neighbors of a node decide not to forward a specific packet and the node will therefore not receive it.

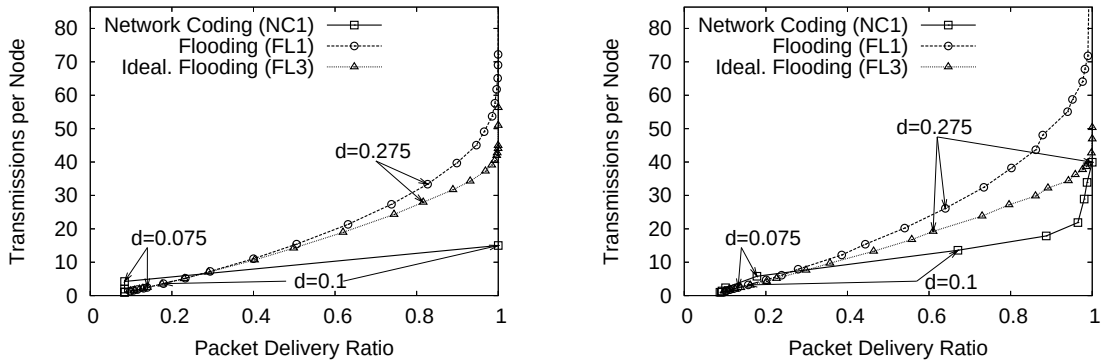

Fig. 3. Transmissions v.s. packet delivery ratio for regular grid (left) and random networks (right) with 144 nodes.

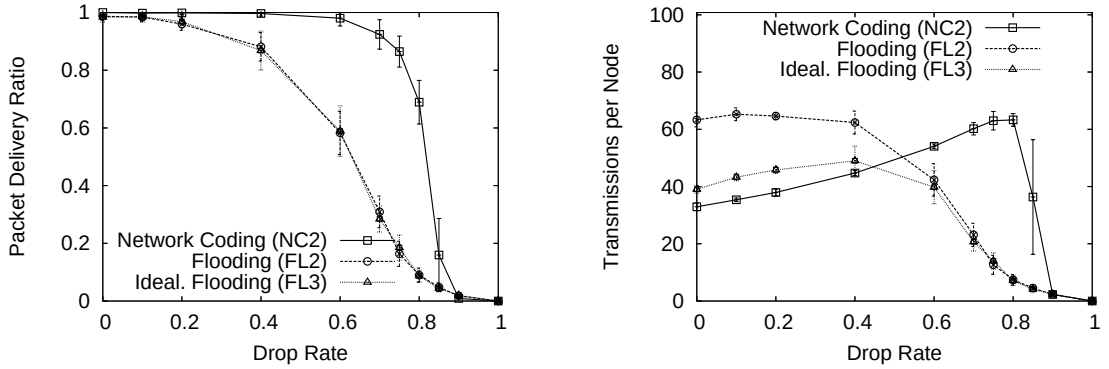

Fig. 4. Packet delivery ratio (left) and number of transmissions (right) for different packet drop rates in a random network with 144 nodes

up to 75%. Both flooding mechanisms have the same packet delivery ratio which drops to 0 significantly earlier than that of network coding. The difference is most apparent for drop rates between 70% and 80%, where network coding achieves delivery ratios that exceed those of flooding by a factor of 5 to 8. The same simulations on a regular grid yield an even higher performance advantage for network coding. Here, the delivery ratio remains at 100% for drop rates of up to 75%.

The graph with protocol overhead shows an increasing number of transmissions T_{NC2} up to a drop rate of about 80%, where the delivery ratio sharply drops to 0. This is due to the fact that more and more packets have to be sent in order to receive 5 non-innovative packets (and thus suppress an own broadcast). In contrast, the number of transmissions of flooding decreases gradually, because the low packet delivery ratio does not leave more packets to be rebroadcasted.

The usefulness of the receive count limit for network coding in random topologies becomes apparent when comparing T_{NC3} with T_{NC4} (again not shown here). For Algorithm *NC3* without receive count limit, T_{NC3} is constant at 72 packets, until it drops of to 0 for packet drop rates higher than 80%. That is, for low drop rates Algorithm *NC4* achieves the same delivery ratio for roughly half the number of transmissions. (The packet delivery ratios do not change significantly.)

From these simulations we can see that network coding indeed outperforms flooding by a very significant margin for similar complexity algorithms and a random MAC layer schedule.

REFERENCES

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

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

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

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

[8] 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.

[9] 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.

[10] 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.

[11] S. Deb and M. Medard, "Algebraic gossip: A network coding approach to optimal multiple rumor mongering," in *Allerton*, Oct. 2004.

[12] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *IEEE Infocom*, 2005.

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

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