

# Performance of Network Coding for Ad Hoc Networks in Realistic Simulation Scenarios

Claudia Campolo

Dip. DIMET, Università Mediterranea di Reggio Calabria  
Email: claudia.campolo@unirc.it

Claudio Casetti, Carla-Fabiana Chiasserini, Saed Tarapiah

Dip. di Elettronica, Politecnico di Torino  
Email: name.lastname@polito.it

**Abstract**—Network coding has recently emerged as an effective solution for multicast and broadcast communications in ad hoc networks. We focus on broadcast traffic and design a network coding-based scheme that we compare against simpler solutions, by using the network simulator ns-2. Indeed, while often the benefits of network coding have been shown via theoretical analysis or in simplified simulation scenarios, our aim is to assess the performance of network coding in ad hoc networks when a realistic MAC protocol is considered. Our results show that the performance of network coding for traffic broadcasting strongly depends on the network node density and on the generation size. In particular, network coding leads to significant gains in terms of end-to-end packet loss probability and protocol overhead only when the average number of neighbors per node is higher than 12.

**Index Terms**—Ad hoc networks, traffic broadcasting, network coding

## I. INTRODUCTION

In ad hoc networks, broadcasting is used for data dissemination with several different aims such as finding a route to a particular node, sending a warning signal or performing service discovery. The simplest method to implement broadcasting is flooding, where every node in the network retransmits a message to its neighbors upon receiving it for the first time. Flooding, however, can lead to undesired effects such as the well-known *broadcast storm problem*: redundant packet retransmissions resulting in repeated contention, collisions, and extra-power consumption. Two typical ways to alleviate the broadcast storm effects consist in reducing possible redundant transmissions and differentiating the timing of retransmissions, as extensively discussed in [1].

In this paper we address the problem of efficiently supporting broadcast traffic in ad hoc networks by using network coding. This technique can indeed reduce the overhead due to multiple copies of broadcast transmissions, by letting intermediate nodes encode multiple packets into a single output packet. Several works, e.g., [2], [3], have shown that network coding can provide very high reliability, bandwidth gains, reduced delays and better traffic distribution, thus leading to both *load balancing* and *energy consumption reduction*. However, previous works have shown these benefits either via theoretical analysis or in simplified simulation scenarios. Here, we evaluate the impact of network coding in a realistic ad hoc network environment with Constant Bit Rate (CBR) traffic, using the network simulator ns-2.

In our scheme, the source node as well as intermediate relay nodes encode native/incoming packets by using random linear network coding and broadcast them to all downstream nodes. We compare the performance of broadcasting based on network coding with that of two simpler schemes, namely flooding-based and deferred broadcasting. Using a realistic MAC layer, we can observe the effects of packets loss and propagation delay on network coding. Indeed, firstly, a single packet loss has the potential to cause multiple packet losses in terms of decoded packets at the receiver, by invalidating the encoded information. Secondly, different propagation delays, typical of multihop communication in ad hoc networks, force the destination to wait for all packets that were encoded together to be received before decoding can be performed. Finally, packetization delay at the source is exacerbated by the need to wait for receiving enough packets from the application layer before encoding can be performed.

The remainder of the paper is organized as follows. In Section II related work is discussed. Section III provides the description of random linear network coding and introduces the notation we adopt. Section IV describes the broadcast schemes we deployed: simple flooding, deferred broadcast and, in more detail, network coding-based broadcasting. We evaluate all the presented broadcast schemes in Section V, by comparing their performances. Finally, Section VI concludes the paper.

## II. RELATED WORK

Several solutions have been proposed with the aim to improve the efficiency of broadcasting in ad hoc networks. A widely applied approach is based on probabilistic message transmissions [1], in which the messages generated by a traffic source are rebroadcast by other nodes with a certain probability. This solution reduces the overhead with respect to simple flooding, however it may lead to a low delivery ratio, especially in the case of scarcely connected nodes. In counter-based schemes [1], instead, a node determines whether to rebroadcast a packet by counting how many identical packets it receives over a given time interval. In distance-based broadcasting [1], nodes make use of their distance from the previous sender as decision criteria. The main idea is to enable those nodes that are located farther away from the sender to rebroadcast the packet, so as to increase the message spatial progress.

Broadcasting techniques have been recently studied also for the dissemination of alarms or warnings in vehicular networks. In [4] broadcast packets include information about the sender position and the message-propagation direction: whenever a node receives a copy of a broadcast message, it rebroadcasts the packet only if the node follows the sender. The solution presented in [5], instead, does not require position-based information.

As for broadcasting based on network coding, a practical scheme is proposed in [6], where network coding is applied only by a subset of nodes which are selected as forwarders. Also, through promiscuous mode, a forwarder can decide not to send a packet if it hears that all of its neighbors have received it. Even though the scheme in [6] gives good performance, it relies on the exchange of neighborhood information and on the partial dominant pruning algorithm presented in [7], which may cause significant overhead and add complexity relatively to probabilistic approaches.

A randomized broadcast scheme based on network coding is proposed in [8]. There, the network coding parameters are finely tuned for the case of a grid topology so as to reduce the packet loss probability with respect to simple flooding. However, such benefits can be perceived only in dense networks where the packet loss probability due to collisions can be higher than the probability of decoding failure. In [9] the impact of several IEEE 802.11-based MAC protocols on the performance of network coding for broadcast communications is analyzed.

Although relevant, we highlight that to the best of our knowledge only a few works, e.g., [4], exist on network coding for ad hoc networks, referring to practical scenarios or using realistic network simulators. Thus, one of the main goals of our work is to assess the performance of network coding for message broadcasting by using the network simulator ns-2 as an evaluation tool. Furthermore, we implement an efficient network coding scheme that does not require nodes to have knowledge on their two-hop neighbors; we compare its performance against both simple flooding and deferred broadcast, the latter including the solution proposed in [5].

### III. RANDOM LINEAR NETWORK CODING: AN OVERVIEW

The network coding scheme we use in this work is the *random linear network coding*, where the output flow at a given node is obtained as a linear combination of its input flows. In more detail, this scheme regards a block of data as a vector over a certain base field and applies a linear combination to this vector before forwarding it. It is referred to as *random* because the transformation is performed by each node independently of the others, and using random coefficients [2]. The coefficients of the combination are typically referred to as *coding vector*, and, by definition, the coefficients are selected independently and randomly from a finite field. We introduce the network coding operations and notations below.

#### A. Network Coding Operations

Three different operations are performed when network coding is applied:

- *Encoding.* Let  $M_1, \dots, M_n$  denote the source native packets, and  $s$  denote the generation size, i.e., the maximum number of native packets that can be encoded together in one new encoded packet. Note that the header of each encoded packet must specify the generation to which the packet belongs.

An encoding node outputs a generic encoded packet:

$$Y_j = \sum_{i=1}^s e_{j,i} M_{j+i-1}$$

with  $e_{j,i} \in GF(2)$ , where  $\mathbf{e}_j = (e_{j,1}, e_{j,2}, \dots, e_{j,s})$  denotes the coding vector independently and randomly chosen at the node to encode the packet. It is worth pointing out that in  $GF(2)$  the sum operation is the bitwise  $\text{xor}$ . The coding vector is included in the header of the transmitted packet and it is used by receivers to decode the data or to further encode the data, as explained below.

- *Re-encoding.* The re-encoding operation is performed over encoded packets. When a node has received  $u \leq s$  encoded packets belonging to the same generation, then the node may generate a new encoded packet, by picking a new random vector  $\mathbf{g}_k = (g_{k,1}, g_{k,2}, \dots, g_{k,u})$ , with  $g_{k,i}$  still chosen independently and randomly, and by computing the linear combination  $X_k = \sum_{i=1}^u g_{k,i} Y_{k+i-1}$  with  $g_{k,i} \in GF(2)$ . It must be noted that the coding vector with respect to the original packets  $M_{k+i-1}, \dots, M_{k+s-1}$  is now  $\mathbf{e}'_k = (e'_{k,1}, e'_{k,2}, \dots, e'_{k,s})$ , where  $e'_{k,i}$  should be computed as the following linear combination:

$$e'_{k,i} = \sum_{h=1}^u g_{k,h} e_{h,i}$$

The new coding vector  $\mathbf{e}'_k$  is included in the header of the newly encoded packet.

- *Decoding.* Decoding at any receiver can be performed by collecting packets of a given generation. These packets yield a system of linear equations that need to be solved to retrieve the original native packets. Suppose a node has received  $v$  encoded packets  $X_1, X_2, \dots, X_v$  belonging to a given generation, with  $v \leq s$  while  $\mathbf{e}'_1, \mathbf{e}'_2, \dots, \mathbf{e}'_v$  represent the coding vectors corresponding to the encoded packets. The decoding matrix  $G$  is as follows:

$$G = \begin{pmatrix} \mathbf{e}'_1 \\ \mathbf{e}'_2 \\ \vdots \\ \mathbf{e}'_v \end{pmatrix}$$

The rank of this matrix is obtained by computing the inverse of  $G$  through the Gaussian elimination method. Let us denote the rank of  $G$  by  $R$ . When the matrix has full rank, i.e.,  $R = v = s$ , for a given generation,

then the node can solve the system of linear equations to retrieve all native packets belonging to that generation. By deriving  $e'_{l,i}$  from the packet headers and solving the linear equations with  $M_i$  as the unknowns:

$$X_l = \sum_{i=1}^v e'_{l,i} M_{l+i-1} \quad l = 1, \dots, v$$

the destination can recover the set of native packets  $M_l, M_{l+1}, \dots, M_{l+v}$ . The receiver can also perform an early decoding when a  $G$  sub-matrix has full rank, i.e., when the sub-matrix rank is equal to  $v$ , with  $v < s$ . In this case, the receiver can recover  $v$  out of  $s$  source native packets belonging to the given generation.

We finally observe that when a node receives a packet, it must check whether it is *innovative* or not, i.e., whether it increases the rank of the decoding matrix  $G$ . If not, the packet is dropped.

#### IV. BROADCASTING SOLUTIONS

Here we introduce our network coding-based scheme and briefly describe two other broadcasting schemes that provide a term of comparison with our own.

##### A. Network Coding-based Broadcasting

Our network coding-based broadcasting is implemented on top of the network layer running over an IEEE 802.11 MAC protocol. This choice allows us to avoid encoding routing address information carried in IP headers.

As suggested in [10], we embed the coding vector in the packet header, thus dispensing with the need for centralized knowledge of the graph topology or decoding functions. In particular, the encoding header follows the IP header as the first piece of data in the IP payload.

The network coding header contains information about the encoded packet, such as: the generation size and identifier, the number of encoded packets along with their size, and the coding vector.

We now detail the different operations performed by the source node, the intermediate nodes, and the receiver nodes. It is worth remarking that, in broadcasting, intermediate nodes are also receiver nodes; however, for the sake of clarity, we distinguish the two operations. Also, note that each node (except for the source) needs to collect  $s$  (recall that  $s$  is the generation size) independent packets for further encoding/decoding, therefore some buffer space is needed at the receiver. In the following, we will call this buffer *NC buffer*.

- *Source node operations.* The source node's application layer generates native packets, which are then encoded at the network coding sub-layer. The source collects  $s$  native packets and encodes them to generate  $s$  different encoded packets  $Y_1, Y_2, \dots, Y_s$ , as explained in Section III. The coding vectors are  $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_s$ : the elements of the generic vector  $\mathbf{e}_i$  ( $i = 1, \dots, s$ ) are all zeros except for the  $i$ -th element, which is equal to one. The source neighbors that receive  $s$  encoded packets belonging to a

given generation, can therefore decode the corresponding  $s$  native packets.

- *Intermediate node operations.* Intermediate nodes perform re-encoding operations. When an intermediate node receives the first encoded packet of a given generation, i.e., its generation identifier differs from the one seen in earlier packets, the packet is cached in the NC buffer and a timer is started (note that one timer per generation is needed). The intermediate node then has to establish whether the subsequently received encoded packets belonging to the same generation are innovative. It thus applies the Gaussian elimination method and checks whether the rank of  $G$  increases when the new encoded packet is added to the buffer. If not, the packet is dropped, otherwise it is cached in the buffer. When  $G$  has full rank, the node performs decoding to retrieve the native packets belonging to the given generation. The native packets are then encoded again into one packet with a random independent coding vector, so that only one packet is forwarded. If the timer of the NC buffer associated to that generation expires before the matrix rank has reached  $s$ , the re-encoding operation is applied to the set of encoded packets buffered at the node and the new encoded packet is forwarded. Therefore, an intermediate node only forwards one re-encoded packet following the reception of at most  $s$  encoded packets.
- *Receiver node operations.* Upon receiving encoded packets, the innovative ones are cached for decoding. After building the decoding matrix  $G$ , its rank is checked. If  $G$  has full rank, then the receiver can recover all native packets in a given generation; an early decoding can be performed when a sub-matrix of  $G$  has full rank  $v$ , with  $v < s$ .

##### B. Simple Flooding

The simple flooding scheme we use as term of comparison is based on the IEEE 802.11 standard. When a node sends a broadcast packet, all its one-hop neighbors will receive it. Since all the neighbors need to rebroadcast the packet in their turn, their transmissions may occur at almost the same time, thus likely resulting in collisions.

To reduce the collision probability, we introduced a simple link-layer modification to the standard: after receiving the broadcast packet, we let the receiving node defer the retransmission by a small time, randomly chosen in the range  $[0, 10]$  ms.

##### C. Deferred Broadcast

The deferred broadcast scheme we consider includes the mechanism proposed in [5], which, for the sake of clarity, we briefly recall.

Upon receiving a broadcast packet, nodes store the information about the predecessor node<sup>1</sup> and, before taking a decision whether to rebroadcast it or not, they wait for a *hold-off* time

<sup>1</sup>The information about the predecessor node is included by each node in the packet.

$T_{ho}$ , during which they listen to possible retransmissions of the same packet by other nodes. In particular, for each packet, the node counts the number of rebroadcast events from other nodes with different predecessors. If this counter exceeds a given threshold, the packet is dropped. We set this threshold to 2, since simulation results show that a greater value does not lead to any improvement. This choice is supported by the findings in [1], where it is stated that if a node rebroadcasts a packet heard more than twice, the extra area it can cover is only about 9%.

Furthermore, in order to ensure that only the farthest nodes from the source rebroadcast packets, we added a further check as an improvement with respect to [5]. Each node receiving a broadcast packet for the first time records the received power level,  $P_{rx}$ . If this value is lower than a certain threshold, which we set equal to twice the receiver sensitivity,  $P_{th}$ , it classifies itself as a *border node* for that packet. A node which hears the same packet rebroadcast from other nodes with different predecessors exactly twice during its hold-off time, forwards the waiting packet only if it is a *border node*, otherwise it drops the packet.

The hold-off time, for which a node waits before a possible rebroadcasting, is computed according to the received signal strength, in such a way that nodes that are farther away from the sender compute a shorter delay and rebroadcast first. The hold-off time is given by:

$$T_{ho} = T_{max} - \frac{P_{rx} - P_{tx}}{P_{th} - P_{tx}} \cdot (T_{max} - T_{min}) - J \quad (1)$$

where  $T_{min}$  and  $T_{max}$  are the minimum and the maximum hold-off time, respectively,  $P_{tx}$  is the transmission power, and  $J \in [0, T_J]$  is a time interval used to prevent concurrent rebroadcasts from nodes that receive a packet with the same power level,  $P_{rx}$ .

For the sake of simplicity, in our simulations we will assume that all nodes have a common radio range and receiver sensitivity; also, we will use the following parameter settings:  $T_{min} = 40$  ms,  $T_{max} = 100$  ms,  $T_J = 30$  ms, and  $P_{tx} = 0.28$  W (i.e., the default value of transmission power in ns-2).

## V. PERFORMANCE EVALUATION

To assess and compare the performance of the three broadcasting schemes discussed above, we implemented them in the network simulator ns-2. We used the two-ray ground model to simulate the propagation on the physical channel, and the 802.11 protocol with a data rate of 1 Mb/s. We consider one source node whose application layer generates CBR traffic; messages have all the same size, equal to 1000 bytes, and the traffic rate is fixed to 50 kb/s. We also experimented with higher bit rates that however yielded exceedingly high values of packet loss probability for the simulated scenarios. Furthermore, since in this work we are not concerned about the delay performance of the proposed network coding-based scheme, we set the maximum buffering timeout of the NC buffer to 1 s.

To make our simulation results comparable to those in [8], we study two different static network scenarios: a grid

topology and a random topology. The grid topology features 225 static nodes deployed within a fixed-size  $100 \times 100$  grid. Instead, the random topology carries 100 nodes uniformly deployed within an area of  $100 \times 100$  m. In the latter scenario, performance metrics are averaged over 10 instances of random network topologies.

The metrics used to evaluate the broadcast schemes are the delivery delay, the packet loss rate, the transmission fairness and the protocol overhead in terms of the ratio of the total number of bytes transmitted at the MAC layer by all nodes (including the source node) to the total number of bytes generated at the source node application layer. The latter metric accounts for the total number of bytes transmitted to broadcast a packet; hence, it can be related to the energy consumed for the broadcast transmissions by the whole network. In particular, in the case of broadcasting based on network coding, by recording the protocol overhead we can compare the reduction in transmitted bytes due to the encoding procedure against the additional bytes needed for the decoding procedure.

In the following, we present the performance of our network coding-based broadcasting for different values of generation size, namely  $s = 2, 3, 4, 5$ , and we compare it with the results obtained through simple flooding and deferred broadcast.

We start by looking at the average value of the packet delivery delay, which is defined as the time elapsed from the time instant when the packet is generated at the source node to the time instant when the packet is received by a destination node. The top plot in Figure 1, i.e., the grid topology results, shows that the delay decreases as the number of neighbors increases (as a result of the increased radio range). The reason for this behavior is that a wider radio range yields fewer hops, hence lower delay. In particular, such a trend can be noticed for each broadcasting scheme under study. In more detail, we find that the values of delay for deferred broadcast are higher than the ones achieved with the simple flooding, due to the hold off time a node (i.e., a node capable of ensuring the most forward progress of the packet) has to wait before transmitting. In the case of random topology (bottom plot), the only remarkable difference is the lower delay experienced by all schemes for small numbers of neighbors (i.e., low connectivity level). Indeed, some of the farther destinations are unreachable and thus they do not contribute to the computation of the average delivery delay.

Next, we focus on the packet loss probability recorded at the application layer. As shown in the top plot of Figure 2, the packet loss probability in the grid topology case decreases as the neighborhood size increases. Indeed, as the network becomes more and more connected, nodes can receive packets from different neighbors thus having more chance to receive all broadcast packets. When compared to flooding and deferred broadcast, however, the broadcast scheme based on network coding suffers higher loss probability than the other schemes, especially in case of large values of generation size and poorly connected networks. Similar conclusions hold for the random topology, with the exception of a higher loss probability for flooding and deferred broadcast in large neighborhoods with

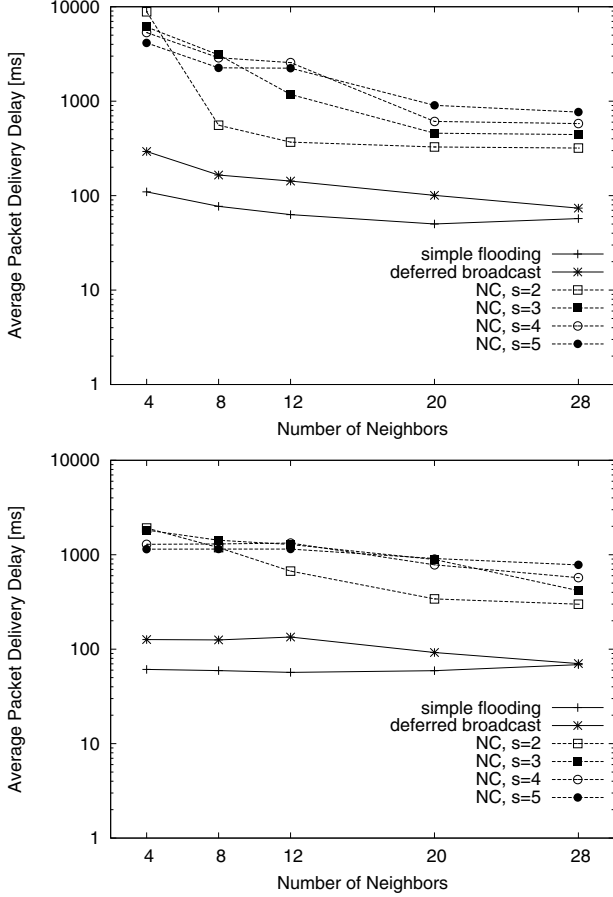


Fig. 1. Average packet delivery delay as a function of the neighborhood size. Grid topology (top) and random topology (bottom)

respect to network coding-based schemes. The reason is to be found in the higher collision probability at the MAC layer: the random node deployment may lead to very dense neighborhood where the medium sharing conditions become extremely crowded. We also remark that our findings differ from those in [8] where it is shown that network coding-based broadcasting significantly outperforms flooding in the case of dense networks with grid topology.

Another metric of interest in wireless networks is the protocol overhead, which is presented in Figure 3. The grid and random topologies yield similar results, although the protocol overhead for the latter is smaller due to the lower connectivity, hence the lower number of transmissions. In the case of simple flooding, the overhead increases with the increase of the node radio range, since the larger the neighborhood size, the larger the number of performed transmissions. Note that, in this case, low values of protocol overhead (i.e., small neighborhood size) correspond to high loss probabilities. When, instead, the deferred broadcast technique is used, the overhead decreases as the node radio range increases. Indeed, for small values of node radio range, the network is scarcely connected and only few nodes can hear each other; it follows that several nodes must take part in the re-broadcasting of traffic packets. As the radio range increases, improving the network connectivity, many nodes refrain from re-broadcasting packets as they hear

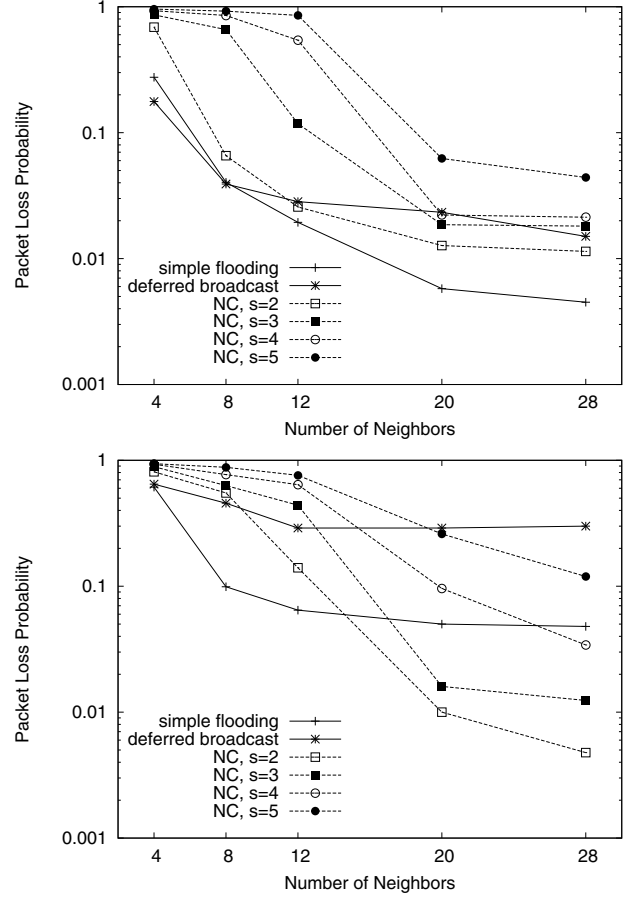


Fig. 2. Average packet loss probability as a function of the neighborhood size. Grid topology (top) and random topology (bottom)

their neighbors transmitting first. However, good performance in terms of overhead corresponds to a lower packet delivery ratio than in the case of simple flooding, as shown in Figure 2. As for network coding-based broadcast, it provides similar results to those achieved by deferred broadcast, although the performance of the network coding mechanism significantly depends on the generation size: the larger the generation size, the more packets are encoded into one, and the smaller the broadcast overhead.

Finally, we want to evaluate the fairness among all nodes (except for the source) in terms of number of transmitted bytes, hence traffic load and energy consumption. To this end, we compute the well-known Jain's fairness index as  $\left(\sum_{i=1}^N x_i\right)^2 / \left(N \sum_{i=1}^N x_i^2\right)$  where  $x_i$  represents the total number of bytes transmitted by node  $i$  for rebroadcasting the received packets, and  $N$  is the number of network nodes not including the traffic source.

By looking at the plots in Figure 4, we observe that, for both topologies, the broadcast schemes based on simple flooding and network coding provide high index values for neighborhood size greater than 4. Worse fairness is recorded for neighborhood size equal to 4 due to lower network connectivity level (some nodes are not receiving nor retransmitting data). Conversely, deferred broadcast yields poor performance

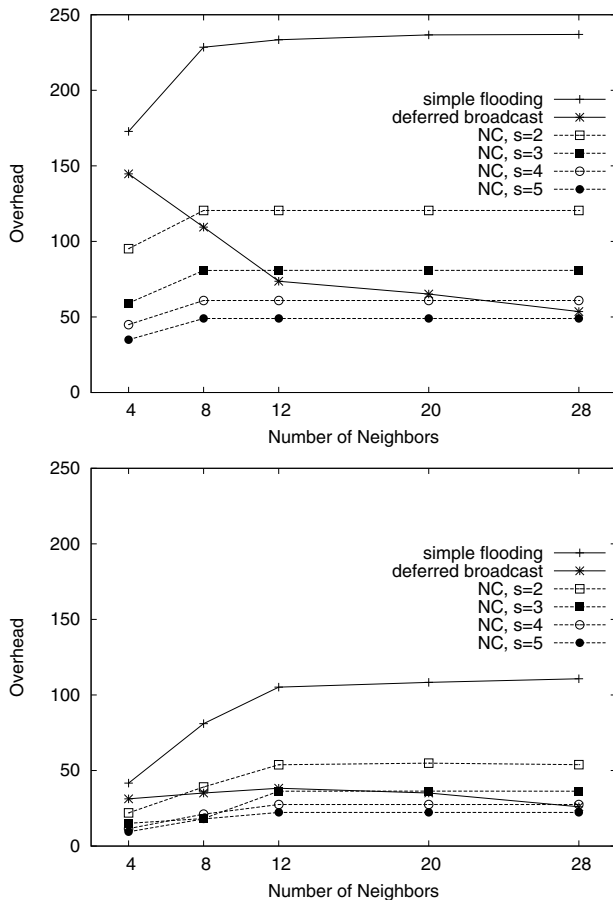


Fig. 3. Protocol overhead as a function of the neighborhood size. Grid topology (top) and random topology (bottom)

in terms of fairness, especially for a large neighborhood size. This is due to the fact that, in deferred broadcast, the same set of nodes are selected as forwarders for all traffic packets, namely the nodes that ensure the greater spatial progress for the packets.

## VI. CONCLUSIONS

We developed a network coding-based scheme for broadcast traffic in ad hoc networks and compared its performance against simpler solutions, based on flooding and deferred broadcast. Simulation results, obtained through the network simulator ns-2, showed that the network node density and the generation size play a crucial role in the performance of network coding for broadcasting. It was also observed that network coding significantly outperforms other broadcasting schemes in terms of end-to-end packet loss probability and protocol overhead only for large neighborhood sizes (i.e., more than 12 neighbors) and generation sizes smaller than or equal to three. Finally, it was shown to achieve a better load balancing among nodes compared to deferred broadcasting.

We remark that previous works on broadcasting based on network coding were carried out either through theoretical or simulative analysis in idealized settings, while our work accounted for the effects of a realistic MAC protocol underneath the coding layer.

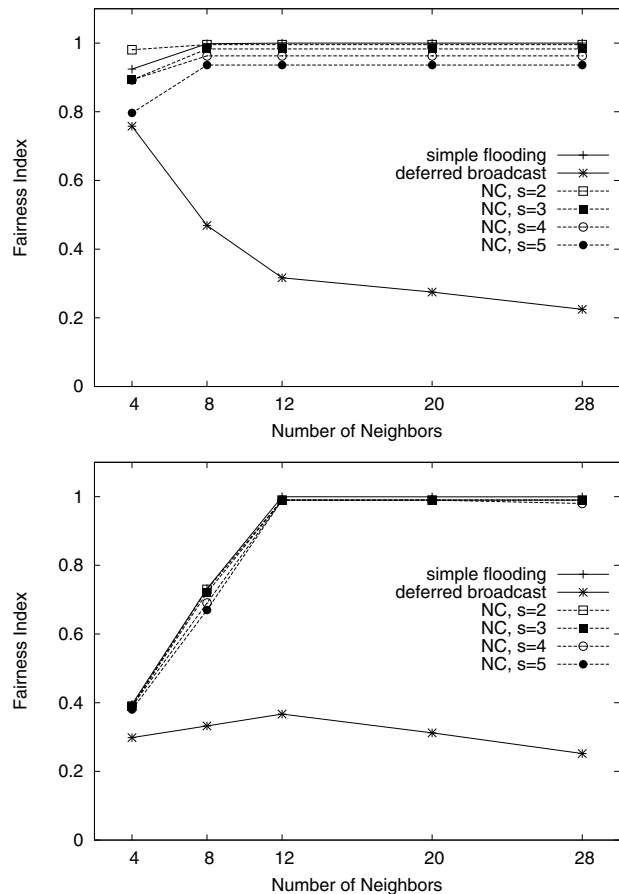


Fig. 4. Fairness index as a function of the neighborhood size. Grid topology (top) and random topology (bottom)

## ACKNOWLEDGEMENTS

This work was supported partially by Regione Piemonte through the VICSUM project and partially by Regione Calabria.

## REFERENCES

- [1] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Networks," in *ACM MobiCom*, Aug. 1999.
- [2] C. Fragouli, J.-H. Le Boudec, and J. Widmer, "Network Coding: An Instant Primer," in *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 63-68, Jan. 2006.
- [3] C. Fragouli, J. Widmer, and J.-I. Le Boudec, "Efficient Broadcasting Using Network Coding," in *IEEE/ACM Trans. on Networking*, vol. 16, no. 2, Apr. 2008.
- [4] E. Fasolo, R. Furiato, and A. Zanella, "Smart Broadcast Algorithm for Inter-Vehicular Communications," in *WPMC*, 2005.
- [5] V. Naumov, R. Baumann, and T. Gross, "An Evaluation of Inter-Vehicle Ad Hoc Networks Based on Realistic Vehicular Traces," in *ACM MobiHoc*, May 2006.
- [6] L. Li, R. Ramjee, M. Buddhikot, and S. Miller, "Network Coding-Based Broadcast in Mobile Ad Hoc Networks," in *IEEE INFOCOM*, 2007.
- [7] W. Lou, and J. Wu, "On Reducing Broadcast Redundancy in Ad Hoc Wireless Networks," in *IEEE Trans. on Mobile Computing*, 2002.
- [8] T. Matsuda, T. Noguchi, and T. Takine, "Broadcasting with Randomized Network Coding in Dense Wireless Ad Hoc Networks," in *IEICE Trans. on Communications*, vol. E91-B, no. 10, pp. 3216-3225, 2008.
- [9] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "On MAC Scheduling and Packet Combination Strategies for Practical Random Network Coding," in *IEEE ICC 2007*, Glasgow, UK, Jun. 2007.
- [10] P.A. Chou, Y. Wu, and K. Jain, "Practical Network Coding," in *Allerton, Conference on Communication, Control, and Computing*, Oct. 2003.