An Empirical Study of Performance Benefits of Network Coding in Multihop Wireless Networks

Dimitrios Koutsonikolas
Y. Charlie Hu
Chih-Chun Wang

TR-ECE-08-11
December 1, 2008

School of Electrical and Computer Engineering
1285 Electrical Engineering Building
Purdue University
West Lafayette, IN 47907-1285

Contents

Abstract

Recently, network coding has gained much popularity and several practical routing schemes have been proposed for wireless mesh networks that exploit interflow network coding for improved throughput. However, the evaluation of these protocols either assumed simple topologies and traffic patterns such as opposite flows along a single chain, or small, dense networks which have ample overhearing of each other's transmissions in addition to many overlapping flows. In this paper, we seek to answer the fundamental question: how much performance benefit from network coding can be expected for general traffic patterns in a moderate-sized wireless mesh network? We approach this question via an empirical study of both coordinated and opportunistic coding based protocols subject to general traffic patterns. Our study shows the performance benefits under both types of coding for general traffic patterns are extremely limited. We then analyze and uncover fundamental reasons for the limited performance benefits.

1 Introduction

Recently, network coding has gained much popularity as a promising technique for improving throughput of routing protocols in wireless mesh networks. With network coding, a router can combine multiple packets within a single transmission, thus making more efficient usage of the network bandwidth. The basic principle of network coding can be easily explained through the 3-node scenario from [11]. If node A wants to send a packet p_A to node B through an intermediate router R and B also wants to send packet p_B to A through the same router R, then R can XOR the two packets and broadcast the encoded packet $p_A \oplus p_B$. Then A can obtain p_B by XORing the encoded packet with its own packet, i.e., $(p_A \oplus p_B) \oplus p_A$, and B can obtain p_A by $(p_A \oplus p_B) \oplus p_B$. This reduces the total number of transmissions from 4 to 3, resulting in a throughput improvement of 4/3.

The work in [11] proposed COPE, a practical protocol that extends this basic principle and it allows nodes to combine more than two packets together by exploiting the broadcast nature of the wireless medium through opportunistic listening. [11] showed a several-fold gain of COPE over a non-coding scheme on a wireless testbed and claims that the maximum gain is unbounded. However, subsequent works [5, 21, 15] identified practical limitations of COPE, and showed that its gain highly depends on the topology, traffic pattern or offered load. Despite these works, the question of how much throughput gain from COPE-style network coding one can expect in general topologies and for generic traffic patterns is still open. Quantifying the practical performance gain from network coding is of timely importance, as it not only helps to guide the design of high performance routing protocols, but also helps to justify the significant research efforts being invested by the community on exploring this new technique.

1.1 Previous Approaches

There have been two bodies of work that tried to quantify the performance gain of network coding in multihop wireless networks. The first one studies theoretical upper bounds of coding gain. The work in [17] showed that the throughput gain in a multihop wireless network with multiple unicast sessions is upper bounded by $2c\sqrt{\pi}\frac{1+\Delta}{\Delta}$ in $2D$ random networks, where Δ is a parameter characterizing the intensity of interference, and $c = max\{2, \sqrt{\Delta^2 + 2\Delta}\}$. The work in [15] quantified the *encoding number*, i.e., the number of packets a node can combine together into a single transmission, showing that in practice it is not unbounded, as [11] suggests, but it is bounded by geometric constraints. It then used this result to provide a tighter upper bound for the throughput gain of network coding equal to $\frac{2n}{n+1}$, where n is the maximum encoding number at any node.

The above theoretical approach only estimates an upper bound for the coding gain, by constructing best-case coding scenarios. The estimated upper bound can be far from achievable for general topologies and traffic patterns. Furthermore, in deriving the upper bound, it is often necessary to make assumptions about a particular coding structure, such as coding opportunities at a hotspot (e.g. [15]) that is crossed by many flows, ignoring the fact that nodes adjacent to the hotspot may also perform network coding which can lead to higher gain.[1]

The second approach (e.g. [5, 21]) identifies that opportunistic coding such as in COPE may, in practice, miss several coding opportunities, depending on the order in which nodes in a neighborhood transmit packets. These works then propose the use of *coordinated* network coding, in which transmissions of neighboring nodes are scheduled with the goal of maximizing the gain from network coding. The work in [5] characterizes the capacity region of a multihop wireless network with a simplified version of COPE, combined with backpressure-based scheduling. However, this work provides only theoretical results, making unrealistic assumptions of a slotted MAC layer, and it is very difficult to implement in practice. The work in [21] proposes a practical coordinated coding protocol built on top of 802.11, which also includes backpressure-based rate control. This work also shows that the theoretical gain (in the absence of packet loss) for two opposite flows traversing a chain of n hops is $\frac{2n}{n+1}$, and the practical protocol can approach this theoretical gain. However, in the evaluation, this work only considers a superficial traffic pattern, consisting of pairs of perfectly overlapping flows going towards opposite directions.

1.2 Our Approach, Roadmap, and Findings

The above discussion suggests that quantifying the *average-case* (rather than providing upper bounds) throughput gain of routing protocols based on network coding under practical scenarios, i.e., in a general topology and under generic traffic patterns, remains an important open question.

Since theoretically analyzing the average-case coding gain is extremely hard, in this paper, we perform an *empirical* study of practical coding gains by subjecting state-of-the-art coding based routing protocols to a general topology and general traffic patterns. In particular, since analyzing the performance of network coding gain under arbitrary topologies is also hard, in this paper we focus on a generic, grid topology, one of the most obvious candidate topologies in a planned deployment of mesh networks.

We carry out our study by first describing the topology, traffic pattern, and coding schemes used in our study, and justifying our choices in Section 2. We then present the overall throughput comparison results among no coding, coordinated coding, and opportunistic coding based routing protocols in Section 3. We then proceed to give an in-depth explanation to the overall throughput comparison in three steps: We first analytically quantify the amount of coding opportunities under a general traffic pattern in Section 4. We then analyze the factors that contribute to the gap between the achieved gain of coordinated and opportunistic coding, respectively, and the theoretical result, in Sections 5 and 6.

Our study shows the performance benefits under both opportunistic and coordinated coding are extremely limited. More importantly, we discover fundamental reasons for the limited performance benefits of network coding: (1) There is an inherent limitation to the coding opportunities for general traffic patterns; (2) Codable flows are typically much longer than non-codable flows and hence suffer low throughput to begin with; (3) Under coordinated coding, the codable flows are squeezed by non-codable flows and hence suffer reduced flow rate, and hence reduced performance benefit from coding; (4) While opportunistic coding appears to capture more coding opportunities, it is also more susceptible to packet loss than coordinated coding and no coding, and this susceptibility can result in lower throughput even compared to no coding.

2 Methodology

We used Glomosim [23] for our empirical evaluation study. Glomosim is a popular wireless network simulator with a detailed physical layer. We used the *2-ray* propagation model. We set the transmission range to 250m; the interference range, based on the physical model parameters used in Glomosim was 460m. To facilitate network coding, we used the 802.11 broadcast MAC protocol with a nominal link data rate of 2Mbps.

[1]One such example is two opposite flows along the same chain of nodes; in that case every node other than the two edge nodes may perform coding operations.

2.1 Topology and Traffic Pattern

We chose a grid node placement for two reasons. First, it is very hard to analyze the more general case of a random deployment. Grid placement offers a controlled environment, where we can easily study the effect of transmission/interference range on our results, and it also helps network coding, creating more opportunities for flows to overlap with each other, compared to, e.g., a hexagon structure. Second, we envision future deployment of community WMNs will largely be planned [18, 4]; such planned deployment not only improves the efficiency and reduces the cost, but also simplifies control, management, and administration of the network. We consider a square area of dimension 1.2Km $\times$ 1.2Km, and place 49 nodes in a grid topology. Based on the selected transmission and interference range we chose, each node can directly communicate with its 4 neighbors along the left, right, up, and down directions. and it is within the interference range of its 2-hop neighbors. In other words, only 3-hop neighbors can transmit simultaneously.

One particular feature of our study is that we consider a *sparse* node deployment, which provides only one next hop for each direction and no redundant node. This deployment does not provide any chance of opportunistic listening, and hence *at most 2* packets can be coded at any node. This choice of deployment is in contrast with previous studies that have assumed *dense* mesh networks (e.g., [2, 1, 11]), and is chosen for two reasons. First, a recent study [3] showed that even low-rate control overhead in non-forwarding links can have a multiplicative throughput degradation on data carrying-links, and this effect worsens as the node density increases, which creates more non-forwarding links. Second, another recent study [19] shows that a node density slightly smaller than ours (about 30 nodes per Km^2) can provide 90% coverage in a currently operational mesh network [24]. The same study argues that client-side solutions (e.g., higher-gain antennas) are much more cost-effective than denser topologies. Note that in our case, 100% coverage is guaranteed due to the grid placement.

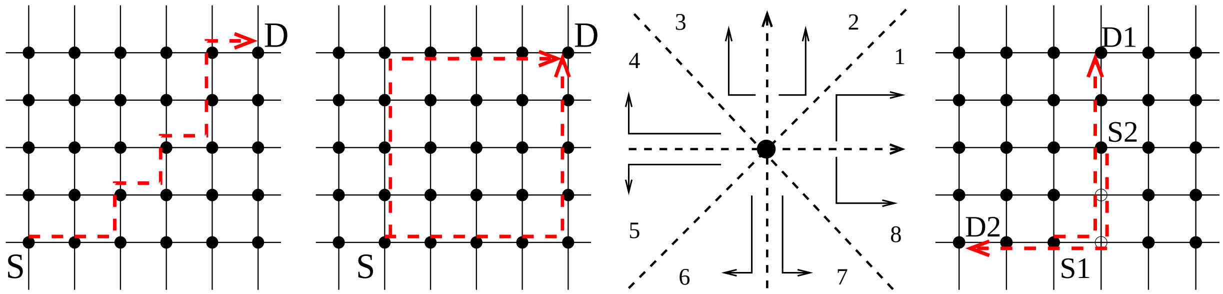
Since in typical deployments of mesh networks all mesh nodes are expected to serve as access routers for some clients, it is expected every mesh node will originate some traffic. In our study, each node randomly selects a destination among the rest 48 nodes and initiates a constant rate flow with a packet size of 1500 bytes for a fixed duration of 1500 sec. We vary the sending rate from 10Kbps up to 2Mbps while keeping it the same for all 49 flows.

Finally, we have also experimented with different packet sizes, different sets of 49 flows, and different grid dimensions (network sizes). The results were qualitatively the same as and quantitatively very similar to those reported in the following sections and are omitted due to the space limitation.

2.2 Routing

We did not use any of the coding-aware routing schemes from the literature for several reasons, since either they are centralized and difficult to realize in practice [22] or they are not deterministic and their control traffic may add variance to the performance of the protocols we examine, depending on the topology and traffic used [14, 8]. In addition, we did not use any link quality metric (e.g. [6]); on a grid structure like the one we used, links are expected to have similar long term average link qualities, however short-term oscillations may cause frequent route changes, which can reduce the amount of existing coding opportunities and also makes the results of our study non-deterministic. Instead, we used fixed, shortest path routing in our study.

There are many possible shortest paths from a source to a destination in a grid (unless the source and the destination have one common coordinate). As we show in Section 3, dimension-ordered routing (DOR) [7], a well-known deterministic routing scheme for mesh and torus interconnection networks for parallel computers, gives more coding opportunities than a randomized shortest-path routing, which always randomly picks either x-axis or y-axis to make forward progress at each hop, shown in Figure 1(a). In DOR, a message is routed in a predetermined order, reducing to zero the offset in one dimension before visiting the next, as shown in Figure 1(b). In a two-dimensional grid, there are two ways of performing DOR for a given flow (source-destination pair). We propose to use a simple heuristic scheme to maximize the coding opportunities of different flows. In the proposed



(a) Random shortest path routing.

(b) Two possible paths with DOR.

(c) Proposed path selection rules to foster coding in DOR: the source chooses the path according to the area that the destination lies in, relative to itself.

(d) Example of path selection for flows $S1 \rightarrow D1$ and $S2 \rightarrow D2$ according to the proposed path selection rules. The partial overlap of the two paths creates 2 coding opportunities (white-filled nodes).

Figure 1. Dimension-ordered routing.

scheme, each node decides which dimension to traverse first based on the relative orientation of the destination, as shown in Figure 1(c). Figure 1(d) shows an example of path selection based on the heuristic of Figure 1(c), where the relative locations of destinations $D1$ (with respect to source $S1$) and $D2$ (with respect to source $S2$) belong to areas 2 and 6, respectively. The three overlapped hops of the two selected paths create two coding opportunities.

2.3 Rate Control

A subtlety in evaluating coding based protocols is that the gain from network coding increases when the offered load in the network is increased, which creates more coding opportunities. For example, in [11], the gain of COPE is maximum for offered load close to 6Mbps, in a network of capacity equal to 6Mbps. However, letting the sources transmit at high data rates without any control can lead the network to congestion, and hence significantly reduced performance, as shown in [16, 13]. For this reason, in our study, we added rate control when comparing different protocols. Since TCP, the de facto reliable transport layer protocol for the wired Internet, has been shown to perform poorly in multihop wireless [9, 12, 10], and interact poorly with network coding [11], we used CXCC [20], a hop-by-hop backpressure-based congestion control protocol.

Briefly, CXCC performs congestion control based on one simple backpressure rule: a node is allowed to transmit a packet only if it has overheard the next hop forwarding the previous packet. This rule, in addition to efficient congestion control, also offers hop-by-hop reliability, since overhearing the downstream node transmitting a packet is an implicit acknowledgment that the node had received it. To deal with packet loss, CXCC uses a small control packet, called Request For Acknowledgment (RFA), to explicitly query the next hop node if a node has not overheard the next hop's transmission for a long time.

2.4 Network Coding

There have been two alternative approaches to developing practical interflow coding-based routing protocols, based on either *opportunistic coding* or *coordinated coding*. We study the performance benefits for both approaches in this paper.

2.4.1 Opportunistic Coding (OpC)

COPE [11] is the first practical implementation of network coding. It performs *opportunistic network coding*, i.e., nodes mix (XOR) packets whenever they get an opportunity to do so, but they never delay packets waiting for coding opportunities to arise. COPE relies heavily on opportunistic listening of all the transmissions in a node's neighborhood, in order to identify coding opportunities, and it also leverages ETX [6] measurements, in order to guess the probability of decoding. In addition, nodes send periodic reception reports to notify their neighbors about which packets they have, and cumulative acknowledgments to improve reliability (COPE is unreliable). All these features add a lot of variance to the protocol's performance.

To include rate control, we used a variation of COPE that is a simple integration of CXCC and COPE (also used in [21]). In this version, CXCC takes care of hop-by-hop reliability, and hence cumulative acknowledgments are not needed.

2.4.2 Coordinated Coding (CoC)

To overcome the fact that COPE and CXCC+COPE do not offer any guarantee that coding opportunities will arise, even in the case where two opposite flows are perfectly overlapping with each other, noCoCo [21] was proposed to perform *coordinated coding* to ensure that nodes will not miss any coding opportunities. noCoCo extends the idea of CXCC for network coding by applying backpressure to two overlapping flows flowing in opposite directions. If a node sends out a coded packet, i.e., an XOR of two packets from two different flows, then from that point on, it is only allowed to transmit coded packets from those two flows. After the transmission of a coded packet, the backpressure is released when the node receives one coded packet from its right neighbor and one from its left neighbor. Those two packets act as implicit acknowledgments for the previous coded packet sent out by the node (one from each side) but they also provide the node with two new packets (one from each flow) which can be mixed together.

noCoCo was designed to fully exploit coding opportunities in perfectly overlapping, bidirectional flows. In a network with many flows, there can be more than two flows that (partially) overlap with each other, and hence multiple ways to perform pair-wise coding. However, noCoCo by itself does not have any mechanism for determining which flows should be coded together. To capture possible maximal coding opportunities, we use an offline centralized greedy heuristic to identify pairs of flows that should be coded together, by iteratively pairing up flows that have the maximum number of coding opportunities. To ensure such identified pairs of flows to be mixed, in our simulation, we start such pairs of flows one after another.

In this paper, we use CXCC, CXCC+COPE, and noCoCo as representative state-of-the-art protocols of **No** **C**oding (NoC), **Op**portunistic **C**oding (OpC), and **Co**ordinated **C**oding (CoC) protocols, respectively. We expect the relative performance of these protocol to generalize to other example protocols. For brevity, we will denote the three protocols simply as NoC, OpC, and CoC in the rest of the paper.

3 Overall Result

Figure 2 shows the total throughput achieved by the 49 flows in our simulation with NoC, OpC, and CoC, as the source sending rate varies. We make the following observations. First, we observe that the performance of all three schemes is similar. The gain of CoC over NoC after the convergence of throughput, i.e., after the source sending rate is higher than 500Kbps, is only 6%–11%. These numbers are much lower than the ones reported in [21], which considered the most ideal environment for network coding where only bidirectional flows were used. One may conjecture that, in a network with random flows, OpC can potentially outperform CoC, since it may identify more coding opportunities, without tying together specific pairs of flows. However, Figure 2 shows that the throughput with OpC is worse than with CoC. In fact, it is even lower than that with NoC (by about 2%–7%).

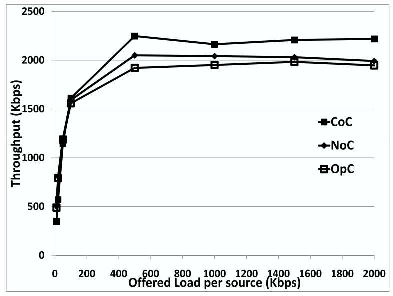


Figure 2. Total throughput for CoC, OpC, and NoC in a 49-node grid.

We contrast our results to those in previous studies. First, our result is different from the one in [21], in which the performance of OpC lies between the performance of NoC and that of CoC. Again, the reason is that bidirectional flows were used in [21] which offered more coding opportunities compared to in our settings. Although some of those opportunities were missed, because of the lack of coordination, the remaining were still enough to increase the throughput compared to NoC. Second, our result is very different from the one reported in [11], where in a random topology and with a random traffic pattern, the original COPE protocol (OpC) offered a 3-4x throughput increase over NoC. This significant gain is due to the small (with a total of 20 nodes) and dense network used in [11], where flow lengths vary between 1 and 6 hops. Such a setting offered high overhearing probabilities, giving nodes the chance to XOR more than 2 packets together. In contrast, our evaluation was done in a larger and sparser network, with dimensions and settings closer to those we believe will be featured by future WMNs. In this network, overhearing does not add any benefit, and the flow lengths are much larger, varying from 1 to 12 hops.

In the rest of the paper, we look inside our overall throughput result to uncover the fundamental reasons why network coding exhibits such low performance benefit for general traffic patterns in a sparse network deployment.

4 Analysis of Coding Opportunities

In this section, we first show that DOR creates coding opportunities that scale well with the (shortest) path length of the flows for general traffic patterns in a large network. We then measure via the simulations the precise expected amount of coding opportunities for such general traffic patterns.

4.1 Analysis

We analyze the asymptotic behavior of the proposed DOR as follows. Consider an $n \times n$ grid. Each of the n^2 nodes sends packets to one uniformly randomly chosen destination. There are thus n^2 flows and let T denote the total number of transmissions required to send one packet for each of the n^2 flows without using network coding. The flows can also be grouped into $0.5n^2$ pairs. As discussed in Figure 1(d), for each pair of flows, the number of coding opportunities is the number of consecutive overlapped hops with the two flows going in the opposite directions minus one. Let S denote the sum of coding opportunities of all pairs. Since different pairing of flows will lead to different S, we use S^* to denote the maximum coding opportunities for any pairing of flows. We then have

Proposition 1 *For any $\epsilon > 0$, the probability that $\{\frac{S^*}{T} \geq (\frac{3}{256} - \epsilon)\}$ approaches one when n tends to infinity.*

Note that as a lower bound, the above proposition implies that even with unidirectional traffic (with random sources and destinations) rather than bidirectional traffic, DOR guarantees a constant, linear fraction of coding opportunities over the total number of transmissions for sufficiently large n.

Proof: Consider a subnetwork of the $n \times n$ grid that consists of 8 $a \times a$ sub-grids as illustrated in Figure 3. For easier reference, we termed the lower left sub-grid the $(1, 1)$ grid and the upper right sub-grid the $(4, 2)$ grid. We choose a

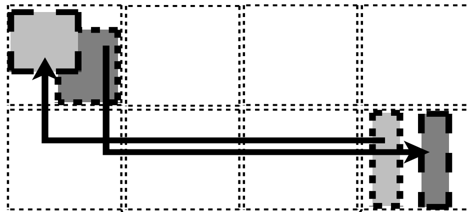


Figure 3. Proof figure: a subnetwork of the $n \times n$ grid that consists of 8 $a \times a$ sub-grids.

to be some constant fraction of n, which tends to infinity as n tends to infinity. The δ_is and ϵ used below are strictly positive constants that can be made arbitrarily close to zero. Consider all nodes in the $(1, 2)$ grid that have their randomly chosen destinations in the $(4, 1)$ grid. With sufficiently large n, there are $a^2\left(\left(\frac{a}{n}\right)^2 - \delta_1\right)$ such nodes with close-to-one probability. We denote these flows as $(1, 2; 4, 1)$ flows. The destinations of these $(1, 2; 4, 1)$ flows are evenly distributed in the $(4, 1)$ grid in the sense that for any y-coordinate, there are $a\left(\left(\frac{a}{n}\right)^2 - \delta_2\right)$ destinations, not necessarily distinct, having the same y-coordinate.

Similarly, there are a^2 source nodes in the $(4, 1)$ grid and $a^2\left(\left(\frac{a}{n}\right)^2 - \delta_3\right)$ of them have destinations in the $(1, 2)$ grid, which are termed the $(4, 1; 1, 2)$ flows. The source nodes of these $(4, 1; 1, 2)$ flows are again evenly distributed such that for any y-coordinate there are $a\left(\left(\frac{a}{n}\right)^2 - \delta_4\right)$ sources having the same y-coordinate. By our rules of DOR (Figure 1(c)), any $(1, 2; 4, 1)$ flow will travel vertically to the bottom first and then horizontally to the right. Similarly, any $(4, 1; 1, 2)$ flow will travel horizontally to the left first and then vertically to the top. We can then pair these two types of flows to introduce coding opportunities. Since for the same y-coordinate, the number of $(1, 2; 4, 1)$ destinations in $(4, 1)$ and the number of $(4, 1; 1, 2)$ sources in $(4, 1)$ are equally matched, we can construct $a\left(\left(\frac{a}{n}\right)^2 - \delta_5\right)$ pairs of flows for any given y-coordinate. Figure 3 illustrates one such pair. This pair of flows will induce at least $2a + 1$ consecutive overlapped hops and thus have at least $2a$ coding opportunities. By summing up the coding opportunities for different y-coordinates, the achievable S by this particular pairing between $(1, 2; 4, 1)$ and $(4, 1; 1, 2)$ flows is $\geq a\left(a\left(\left(\frac{a}{n}\right)^2 - \delta_5\right)\right)2a$. By choosing $a = n/4$ (since $4a$ must be no larger than n), we have that with close-to-one probability, the pairing between $(1, 2; 4, 1)$ and $(4, 1; 1, 2)$ flows will lead to $\left(\frac{1}{512} - \delta_5\right)n^3$ coding opportunities. We can also consider 12 similar pairings of flows such as $(1, 2 + k; 4, 1 + k)$ versus $(4, 1 + k; 1, 2 + k)$, $(1, 1 + k; 4, 2 + k)$ versus $(4, 2 + k; 1, 1 + k)$, $(1 + k, 1; 2 + k, 4)$ versus $(2 + k, 4; 1 + k, 1)$, and $(2 + k, 1; 1 + k, 4)$ versus $(1 + k, 4; 2 + k, 1)$ for $k = 0, 1, 2$. As a result, the S of the above pairing is no less than $(\frac{12}{512} - \delta_6)n^3$. Since the maximum transmission for each flow is $2n - 2$, the total number of transmissions T is $\leq n^2(2n - 2) < 2n^3$. Jointly, we have that the probability $\left\{\frac{S}{T} > \left(\frac{3}{256}\right) - \epsilon\right\}$ tends to one when n tends to infinity. Since the optimal pairing will lead to an $S^* \geq S$, the proof is complete. $\square$

Note that DOR is a critical component for achieving this linear number of coding opportunities for each pair of flows. Let (x_s, y_s) and (x_d, y_d) denote the coordinates of the source and destination. *DOR chooses a unique route along the edges of the rectangle spanned by* (x_s, y_s) *and* (x_d, y_d), which greatly enhances the number of coding opportunities. If a shortest path is chosen randomly without using DOR, then the path from (x_s, y_s) to (x_d, y_d) will zig-zag within the rectangle, which is unlikely to result in *consecutive overlapped hops* (see Figures 1(a), 1(b)).

4.2 Validation of Theoretical Result

The analysis above shows DOR guarantees coding opportunities asymptotically proportional to the flow lengths. To measure the precise coding gain under general traffic patterns, i.e., each source initiates a flow to a random destination, we resort to a simple simulator that assumes an ideal MAC and physical layer without contention or

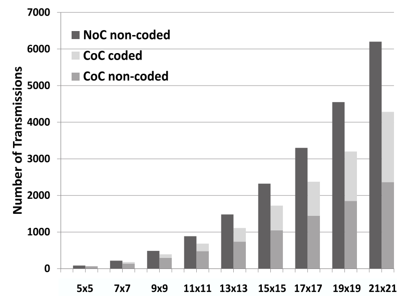


Figure 4. Total number of non-coded and coded transmissions with NoC and CoC when each source sends one packet (simplified simulator) for different grid sizes. Results for each size are averages over 10 different sets of flows.

packet losses, to count the total number of transmissions required in order to deliver one packet from each source to its corresponding destination with and without coding. With coding, we again used our greedy heuristic to identify pairs of flows that should be mixed together, in order to maximize the coding opportunities.

Figure 4 shows the total number of transmissions without coding and the total number of coded and non-coded transmissions with coding as the grid size varies, assuming all coding opportunities are captured by the routing protocol. The results are average over 10 different sets of flows. It confirms that coding opportunities scale proportionally with the average flow length.[2] In particular, the percentage of coded transmissions with CoC over the total number of transmissions with NoC increases from 14% on average in a 5x5 grid up to 31% on average in a 21x21 dimension grid, far better than the lower bound given in Proposition 1. Nevertheless, for practical network sizes (less than 100 nodes), the number of coding opportunities and hence the expected throughput improvement are rather small (less than 23%).

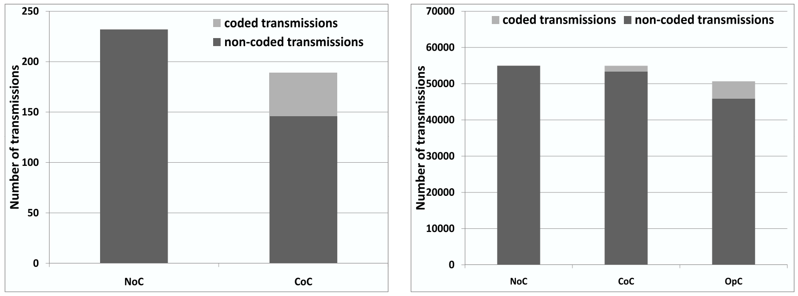
4.3 Comparison of Theoretical and Practical Results

We now return to the set of the 49 flows used in the simulation study in Section 3 and compare the total number of non-coded and coded transmissions obtained in theory and in practice. Figure 5(a) shows the predicted (from Section 4.2) total number of transmissions for that set of flows without (NoC) and with (CoC) coding. We observe that the total number of transmissions required to deliver one packet from each source to its corresponding destination is reduced from 232 with NoC to 189 with CoC, a reduction of 22%, i.e., there are $232 - 189 = 43$ coded transmissions.

However, the reduction in the total number of transmissions from simulations is much worse than the above predicted values. Figure 5(b) shows, compared to NoC, CoC and OpC reduce the total number of transmissions by only 0.5% and 7.8%, respectively, much lower than the predicted 22% reduction. These small reductions in transmissions resulted in 11% higher throughput for CoC but 2% lower throughput for OpC, compared to NoC, as shown in Figures 2. Note that we cannot make any direct correlation between the reduction in the total transmissions and the total throughput improvement, since the 49 flows under consideration have different lengths; reduction in the number of transmissions only gives an indication of possible throughput improvement.

In summary, we saw that for general traffic patterns, the theoretically predicted coding opportunities and hence potential reduction on the number of transmissions with network coding is already small. However, the comparison of the theoretical result with the practical one raised a number of intriguing questions. Q1: Why is the number of coding opportunities captured in practice even lower than the theoretically predicted number? Q2: When flows follow a random traffic pattern, why does OpC discover more coding opportunities than CoC, although the latter

[2]The scaling is actually slightly faster than linear, due to the discretization effect for small grid sizes.



(a) Result from the simplified simulator – each source sends one packet.

(b) Result from Glomosim – each source sends packets at 2Mbps.

Figure 5. Total number of non-coded and coded transmissions with NoC, CoC, and OpC for the 49-node scenario of Figure 2 with the simplified simulator and Glomosim.

is guaranteed to capture all the coding opportunities of pair-wise overlapping flows? Q3: Why is the throughput with CoC higher than the throughput with OpC although OpC discovers more coding opportunities than CoC? In particular, why is throughput with OpC even lower than with NoC?

In the next two sections, we analyze separately CoC and OpC in detail, to find the answers to the above questions.

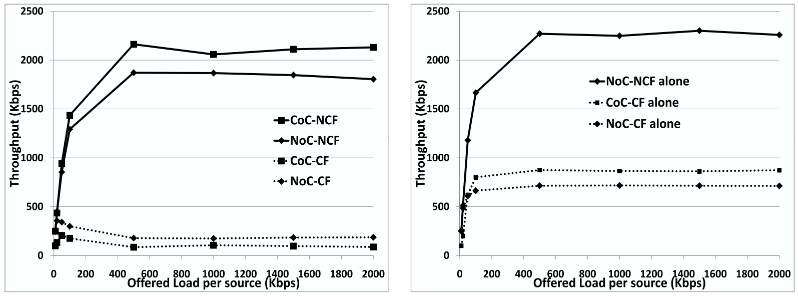
5 Coordinated Coding

We begin with coordinated coding (CoC).

Separating codable and non-codable flows. In trying to answer the above questions, we take a closer look at the 49 flows by breaking them into two classes. Flows that had their packets coded with packets from other flows at one node at least belong to class CF (codable flows). All the remaining flows, which had all their packets transmitted non-coded belong to class NCF (non-codable flows). Flows of class CF are the ones that contribute to the reduction of the total number of transmissions due to coding. With CoC, once a node decides to mix together two packets from two CF flows, it will mix together *all* the subsequent packets belonging to those two flows. We found that out of the 49 flows we initiated, only 24 belong to class CF, and the remaining 25 belong to class NCF. These numbers show again the low number of coding opportunities, since half of the flows could not be mixed with any other flow.

We repeated the experiment of Section 3, however, this time we measured separately the total throughput of the 24 CF flows and the 25 NCF flows. The results are shown in Figure 6(a). We observe that the 25 NCF flows achieve much higher throughput than the 24 CF flows, both with NoC and with CoC. More surprising, the throughput of the 24 CF flows is less with CoC than with NoC. In other words, the use of network coding has a "squeezing effect" on the flows that actually have their packets mixed together, reducing the total throughput of those flows by 40% - 53% in the steady state (i.e., for offered load per source higher than 500Kbps), while the throughput of the remaining flows, whose packets are never mixed together, increases by 10% - 18%. In fact, these 25 NCF flows utilize almost the whole network capacity, since their total throughput is one order of magnitude larger than the throughput of the 24 CF flows. This explains why the total throughput with CoC is higher than with NoC in spite of the small number of coding opportunities, and in spite of the fact that the *relative* throughput decrease for the 24 CF flows is much larger than the relative throughput increase of the 25 NCF flows.

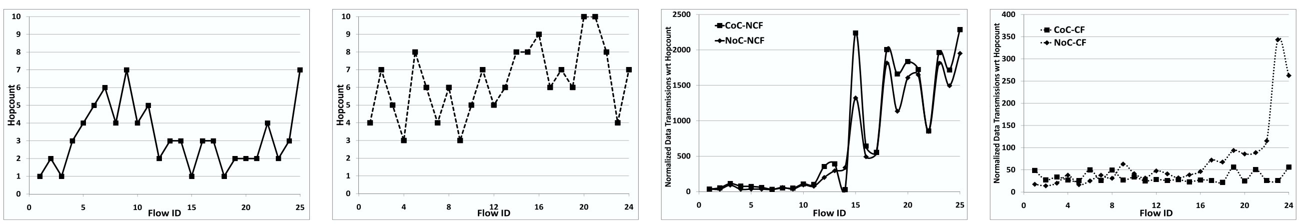
To verify that this counter-intuitive result is not due to some artifact in the protocol implementation, we repeated the simulation for CoC with only the 24 CF flows present in network. We also repeated the simulation for



(a) Total throughput for the 24 CF flows and 25 NCF flows with CoC and NoC when they coexist in a 49-node grid.

(b) Total throughput for the 24 CF flows alone and the 25 NCF flows alone with CoC and NoC in a 49-node grid.

Figure 6. Throughput comparison of CF and NCF flows with CoC and NoC, when they coexist and when they are alone, in a 49-node grid.



(a) Hopcounts for each of the 25 NCF flows.

(b) Hopcounts for each of the 24 CF flows.

(c) Normalized data transmissions for each of the 25 NCF flows with CoC and NoC.

(d) Normalized data transmissions for each of the 24 CF flows with CoC and NoC.

Figure 7. Hopcounts and normalized data transmissions for non-coded and coded flows with CoC and NoC.

NoC twice, first only with the 24 CF flows, and then only with the 25 NCF flows.[3] The results are shown in Figure 6(b). We observe that the results for the 24 CF flows are reversed, compared to Figure 6(a); when only the 24 CF flows are present in the network, the throughput with CoC is 21% higher than with NoC. This confirms that coding indeed helps when all flows are codable. It is the interaction of CF vs. NCF flows that causes the counter-intuitive result in Figure 6(a)! Note that all the results in [21] were obtained from settings similar to the one in Figure 6(b), with only CF flows present in the network.

Correlating with flow lengths. In Figure 6(b), we also observe that the throughput of the 25 NCF flows, without the presence of the 24 CF flows, is more than 2.5 and 3.2 times higher than the throughput of the 24 CF flows with network coding and without network coding, respectively. To understand this, we correlated the total number of transmissions of packets from each of the 49 flows with the lengths (in terms of hopcounts) of the flows, when all of them are simultaneously present in the network.

[3] The result for CoC with only the 25 NCF flows present is exactly the same as the result with NoC, since there is no coding opportunity (i.e., no overlap) among those flows.

Figures 7(a), 7(b) show that the 25 NCF flows are on average much shorter than the 24 CF flows. The hopcounts for the NCF flows range from 1 to 7, with an average hopcount of 3.2, while for the CF flows they vary from 3 to 10, with an average hopcount of 6.3. This is intuitive, since the longer the two flows, the higher the probability that they will overlap (at least partially), creating some coding opportunities.

Figures 7(c),7(d) plot the normalized number of data transmissions for a flow with respect to its hopcount, which gives a clear indication of the corresponding throughput for that flow. To make this calculation correct, each coded packets is counted twice, once for each flow. These figures show again that the normalized number of transmissions (i.e. throughput) for the NCF flows are in general much larger than those for the CF flows, and again these numbers for most of the NCF flows are higher with CoC than with NoC, but lower for most of the CF flows, which explains the throughput results in Figures 2 and 6(a).

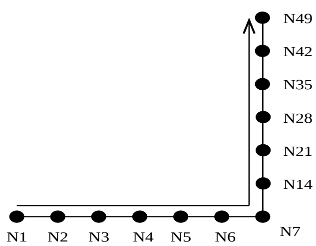
5.1 Microbenchmarks

We have seen that with network coding, the throughput of CF flows decreases, while the throughput of NCF flows increases, and we have correlated this initially counter-intuitive result to the lengths of the flows. To better understand this correlation, we developed 3 microbenchmarks, using toy topologies taken from the 49-node grid topology, as shown in Figure 8.

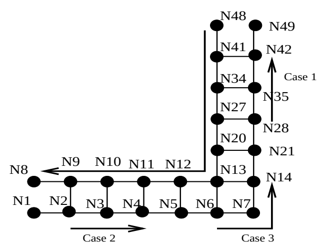
Microbenchmark 1: Throughput as a function of flow length. With this microbenchmark, we use a single flow to study the throughput of a flow as a function of the flow length. We used node $N1$ as the source and we change the destination, traveling along the outer row and column of the 49-node grid, as shown in Figure 8(a), thus varying the flow length from 1-12 hops. Figure 8(d) shows the throughput initially decreases rapidly as the hopcount increases, but it finally stabilizes, due to spatial reuse. Since NCF flows are in general much shorter than CF flows, it is expected to achieve higher throughput. However, in Section 5 we saw that NCF flows had an average length of 3.2 hops, and CF flows had an average length of 6.3 hops – hence, according to Figure 8(d), their throughput difference should be about 23%. However, the actual throughput gap was much larger as we observed in Figure 6(a).

Microbenchmark 2: A long and a short NCF flow competing with each other. With this microbenchmark, we study the interaction between a long and a short flow, when they are competing with each other, i.e., they are in the interference range of each other. We use a 10-hop flow from node $N48$ to node $N8$, and a 2-hop flow parallel to the long one, as shown in Figure 8(b). We repeat the simulation three times, changing the location of the short flow: from node $N28$ to node $N42$, near the source of the long flow (case 1), from node $N2$ to node $N4$, near the destination of the long flow (case 2), and from node $N6$ to node $N14$, in the middle of the long flow (case 3). The results for all three cases are shown in Figure 8(e). Since the two flows are in the interference range of each other, ideally they should share the bandwidth bottleneck and get equal throughput. However, we observe that in all 3 cases, the short flow outperforms the long one, by 16%, 32%, and 148%, respectively. Hence, we observe that short flows win the competition against long flows, getting a larger fraction of the available bandwidth, and the impact is larger when the short flow is located near the middle of the long flow.

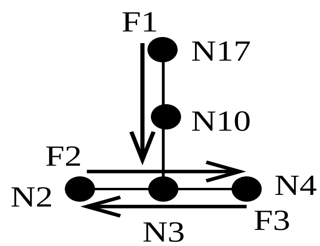
Microbenchmark 3: Two CF flows competing with an NCF flow of equal length. We now study the interaction between CF flows and NCF flows, when they are in the interference range of each other. We want to study this interaction independently of the flow lengths, hence we used three flows of the same length, as shown in Figure 8(f). We initiate two perfectly overlapping flows, traveling towards opposite directions, $F2$ from node $N2$ to node $N4$, and $F3$ from node $N4$ to node $N2$, which create the well-known *Jon-and-Dina* scenario from [11]. The intermediate node $N3$ can mix together packets from these two flows, and CoC guarantees that it does not miss any coding opportunity, i.e., it always has one packet from each flow available, and it transmits only encoded packets. We initiate another flow $F1$, from node $N17$ to node $N3$. This is an NCF flow, since its packets can never be mixed with the packets of any of the other two flows. We repeat the experiment twice, once with NoC, and once with CoC. Since all three flows have the same length, and all nodes are in the interference range of each other, they should ideally get all the same throughput. However, Figure 8(c) shows with both schemes, the NCF



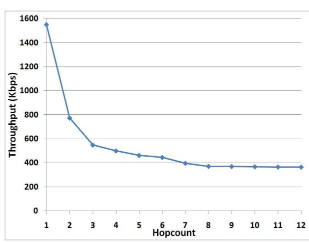
(a) Microbenchmark 1: Throughput as a function of flow length.



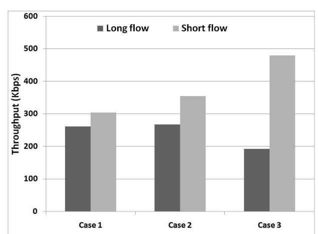
(b) Microbenchmark 2: A long and a short NCF flow competing with each other.



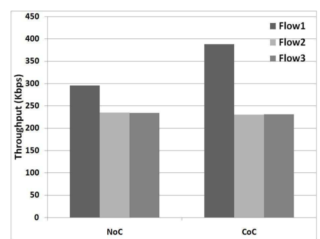
(c) Microbenchmark 3: Two CF flows competing with an NCF flow of equal length.



(d) Microbenchmark 1 result: Throughput decreases with flow length.

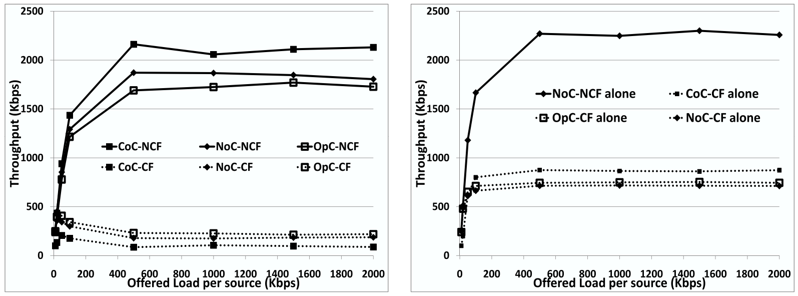


(e) Microbenchmark 2 result: A shorter NCF flow "squeezes" a longer MC flow.



(f) Microbenchmark 3: Two CF flows are "squeezed" by an NCF flow of equal length.

Figure 8. 3 microbenchmarks explaining the performance difference between CF and NCF flows with CoC and NoC.



(a) Total throughput for the 24 CF flows and 25 NCF flows with CoC, NoC, and OpC when they coexist in a 49-node grid.

(b) Total throughput for the 24 CF flows alone and the 25 NCF flows alone with CoC, NoC, and OpC in a 49-node grid.

Figure 9. Throughput comparison of CF and NCF flows with CoC, NoC, and OpC when they coexist and when they are alone in a 49-node grid.

flow $F1$ gets higher throughput than the two CF flows. Also, when we move from NoC to CoC, throughput of each of the two CF flows, which now have their packets coded together at node $N3$, is reduced by a small percentage of 2%, although according to [11], it should increase by 33% (since the total number of transmissions required for these two flows is reduced from 4 to 3). On the other hand, the throughput of the NCF flow increases by 32%. We thus see that, although the number of transmissions required for the two CF flows is reduced, this reduction is not translated to a throughput improvement. The reason is that with coordinated coding, these two flows are tied to each other, and node $N3$ cannot proceed unless it has a new packet from each of them. The two CF flows have to compete not only against each other, but also against the NCF flow. Even if one of them, say $F2$, wins the competition, and sends a packet to $N3$, $N3$ cannot forward the packet until it also receives a packet from the other CF flow. And $F2$ cannot compete again for the channel due to backpressure, unless it receives a coded packet from $N3$. This slack favors the NCF flow $F1$, which experiences less contention, and can send packets at a faster rate.

6 Non-coordinated coding

We now return to OpC. In Figure 5(b), we saw that OpC had a larger fraction of coded over non-coded transmissions, compared to CoC. Yet it has the lowest total number of total transmissions, and in Figure 2 the lowest total throughput, among the three schemes. To explain this result, in Figure 9(a) we plot separately the throughput with OpC for the 25 NCF flows and the 24 CF flows. We also keep the same breakdown for NoC and CoC from Figure 6(a) for easier comparison.

Figure 9(a) shows that OpC exhibits exactly the opposite behavior compared to CoC. With OpC, the throughput for the 24 CF flows increases compared to NoC, while it decreases for the 25 NCF flows. Actually, the throughput with OpC for the 24 CF flows is the highest among the three schemes, and on the other hand, the throughput with OpC for the 25 NCF flows is the lowest among the three schemes. Consistent with this observation is our finding that the number of data transmissions per flow with OpC drops for most of the NCF flows, compared to with NoC, while it increases for most of the CF flows, i.e., the trend is opposite from with CoC (we omit the graphs due to space limitation). This suggests that CF flows have some "squeezing effect" on NCF flows, although not very strong, since NCF flows still achieve much higher throughput than CF flows.

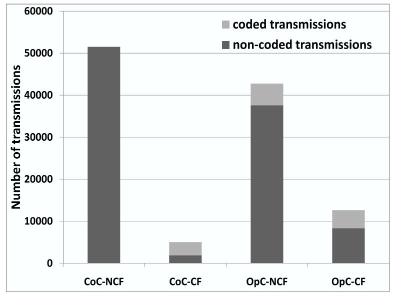


Figure 10. Total number of non-coded and coded transmissions for the 25 NCF flows and the 24 CF flows with CoC and OpC with a source sending rate of 2Mbps.

OpC misses coding opportunities between CF flows (alone). To understand coding opportunities captured by OpC, we first repeated the simulation with OpC for the 24 CF flows defined in Section 5 *alone*. The result is shown in Figure 9(b), in which we have also added the curves from Figure 6(b). for easier comparison. We observe the throughput of the 24 CF flows with OpC is only slightly better than with NoC (less than 5% difference), and lower than with CoC (10-15% difference). This suggests that in a network with only CF flows, which exhibit many coding opportunities, OpC actually misses many of them due to lack of coordination, resulting in lower performance than CoC.

OpC finds coding opportunities between CF and NCF flows. Actually, since OpC applies only one-way backpressure to individual flows and it never ties two flows together after XORing one packet from each of them, the classification of CF and NCF flows, which we carried over from Section 5, has no meaning for OpC. In other words, OpC can potentially mix packets from CF flows with NCF flows. Figure 10 shows the breakdown of the total number of transmissions with CoC and OpC, separated for the 24 CF and 25 NCF flows (according to the classification based on CoC), into coded and non-coded ones, for source sending rate equal to 2Mbps. Again each coded packet is counted twice, once for each flow.

We make two observations from Figure 10. (1) The 4th bar, which shows the total coded transmissions under OpC, is much higher than the 2nd bar, for the total coded transmissions under CoC, suggesting in total OpC captures more coding opportunities. (2) With OpC, the majority of coding opportunities come from coding between CF flows and NCF flows, as shown by comparing the coded portion of the 3rd and the 4th bars. Hence we observe a tradeoff with OpC compared to CoC; OpC misses many coding opportunities among codable flows due to lack of coordination, but, on the other hand, it captures more coding opportunities between random pairs of flows

Why is the total throughput with OpC the lowest among the three schemes, lower even than with NoC? To understand the reason for this fundamental question, we plot again in Figure 11 the total number of transmissions with each scheme, and their breakdown into coded and non-coded ones, as in Figure 5(b), but this time we also count retransmissions (all previous figures that showed number of transmissions did not include retransmissions). We observe that when retransmissions are also counted, OpC makes the largest number of transmissions, followed by NoC, and then by CoC. Also, the increase in the total number of transmissions when counting retransmissions is 13% for NoC, 12.5% for CoC, and 23.5% for OpC. This implies that OpC is the most sensitive to packet failures among the three schemes, and it is forced to make many retransmissions. This explains why OpC achieves the lowest throughput and CoC achieves the highest, among the three schemes.

Although the difference in the total number of transmissions including retransmissions is less than 2%, this difference has a larger impact on throughput difference (CoC has 6-11% higher throughput than NoC, which in turn has 2-7% higher throughput than OpC, as we saw in Figure 2), for the following reasons. (1) Retransmissions are incurred after exchange of various types of control packets. OpC, which suffers most from packet losses, also

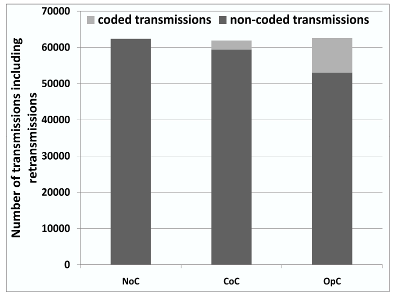


Figure 11. Total number of non-coded and coded transmissions, including retransmissions, with NoC, CoC, and OpC, for the 49-node scenario of Figure 2.

transmits the largest number of control packets, which also reduce the available bandwidth for data transmissions. (2) If we look at the ratio of coded transmissions with retransmissions over coded transmissions without retransmissions, for each of the two network coding schemes, we observe this ratio to be 1.53 for CoC, and 1.99 for OpC. Hence, coded transmissions with OpC are more sensitive to packet losses than with CoC, and the large number of retransmissions required for those transmissions cancels a large fraction of the benefit of network coding.

The combined effect is that the throughput of CF flows with OpC does not increase as much as with CoC, because of the increased number of retransmissions and control packets, and on the other hand, the throughput of NCF flows is reduced with OpC, also due to the increase in the number of retransmissions, and the squeezing from CF flows. Since throughput of the long CF flows is much lower than throughput of the short NCF flows to begin with, the small increase of the former is not enough to compensate for the decrease of the latter, and the overall result is reduced throughput with OpC compared to NoC.

7 Conclusion

In this paper, we presented an empirical study of the *average-case* performance gain from adding network coding to routing protocols in wireless mesh networks. Our study shows the performance benefits under both coordinated and opportunistic coding for general traffic patterns are extremely limited. More importantly, we investigated and uncovered the fundamental reasons for the limited performance benefits: (1) There is an inherent limitation to the coding opportunities for general traffic patterns; (2) Codable flows are typically much longer than non-codable flows and hence suffer low throughput to begin with; (3) Under coordinated coding, the codable flows are squeezed by non-codable flows and hence suffer reduced flow rate, i.e., reduced performance benefit from coding; (4) While opportunistic coding appears to capture more coding opportunities, it is also more susceptible to packet loss than coordinated coding and no coding, and this susceptibility can result in lower throughput even compared to no coding.

Acknowledgment

This work was supported in part by NSF grant CNS-0626703.

References

[1] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. In *Proc. of ACM SIGCOMM*, 2004.

[2] John Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *Proc. of ACM MobiCom*, 2005.

[3] Joseph Camp, Vincenzo Mancuso, Omer Gurewitz, and Edward Knightly. A measurement study of multiplicative overhead effects in wireless networks. In *Proc. of IEEE INFOCOM*, 2008.

[4] Joseph Camp, Joshua Robinson, Christopher Steger, and Edward Knightly. Measurement driven deployment of a two-tier urban mesh access network. In *Proc. of ACM MobiSys*, 2006.

[5] Prasanna Chaporkar and Alexandre Proutiere. Adaptive network coding and scheduling for maximizing throughput in wireless networks. In *Proc. of ACM International MobiCom*, 2007.

[6] Douglas S. J. De Couto, Daniel Aguayo, John C. Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, 2003.

[7] W.J. Dally and C.L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. on Computers*, 36(5):547–553, May 1987.

[8] Saumitra M. Das, Yunnan Wu, Ranveer Chandra, and Y. Charlie Hu. Context based routing: Technique, applications and experience. In *Proc. of USENIX NSDI*, April 2008.

[9] Zhenghua Fu, Petros Zerfos, Haiyun Luo, Songwu Lu, Lixia Zhang, and Mario Gerla. The impact of multihop wireless channel on TCP throughput and loss. In *IEEE Infocom*, 2003.

[10] Abhinav Gupta, Wormsbecker, and Carey Williamson. Experimental evaluation of TCP performance in multi-hop wireless ad hoc networks. In *Proc. of MASCOTS*, 2004.

[11] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Medard, and Jon Crowcroft. Xors in the air: Practical wireless network coding. In *Proc. of ACM SIGCOMM*, August 2006.

[12] Vikas Kawadia and P.R. Kumar. Experimental investigations into TCP performance over wireless multihop networks. In *SIGCOMM E-WIND Workshop*, 2005.

[13] Dimitrios Koutsonikolas, Y. Charlie Hu, and Konstantina Papagiannaki. How to evaluate exotic wireless routing protocols? In *Proc. of ACM HotNets-VII*, 2008.

[14] Jilin Le, John Chi Shing Lui, and Dah Ming Chiu. Dcar: Distributed coding-aware routing in wireless networks. In *Proc. of IEEE ICDCS*, 2008.

[15] Jilin Le, John Chi Shing Lui, and Dah Ming Chiu. How many packets can we encode? – an analysis of practical wireless network coding. In *Proc. of IEEE INFOCOM*, 2008.

[16] Yi Li, Lili Qiu, Yin Zhang, Ratul Mahajan, Zifei Zhong, Gaurav Deshpande, and Eric Rozner. Effects of interference on throughput of wireless mesh networks: Pathologies and a preliminary solution. In *Proc. of HotNets-VI*, 2007.

[17] Junning Liu, Dennis Goeckel, and Don Towsley. Bounds of the gain of network coding and broadcasting in wireless networks. In *Proc. of IEEE INFOCOM*, 2007.

[18] Joshua Robinson and Edward Knightly. A performance study of deployment factors in wireless mesh networks. In *Proc. of IEEE INFOCOM*, 2007.

[19] Joshua Robinson, Ram Swaminathan, and Edward Knightly. Assessment of urban-scale wireless networks with a small number of measurements. In *Proc. of ACM MobiCom*, 2008.

[20] Bjorn Scheuermann, Christian Lochert, and Martin Mauve. Implicit hop-by-hop congestion control in wireless multihop networks. *Elsevier Ad Hoc Networks*, 6(2):260–286, April 2008.

[21] Bjrn Scheuermann, Wenjun Hu, and Jon Crowcroft. Near-optimal coordinated coding in wireless multihop networks. In *Proc. of CoNEXT*, 2007.

[22] Sudipta Sengupta, Shravan Rayanchu, and Suman Banerjee. An analysis of wireless network coding for unicast sessions: The case for coding-aware routing. In *Proc. of IEEE INFOCOM*, 2007.

[23] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. Glomosim: A library for parallel simulation of large-scale wireless networks. In *Proc. of PADS Workshop*, May 1998.

[24] Technology For All (TFA). `http://tfa.rice.edu`.