Optimal Opportunistic Routing and Network Coding for Bidirectional Wireless Flows

Tahir Mehmood* Lavy Libman* Hooman Reisi Dehkordi† Sanjay K. Jha*
*School of Computer Science and Engineering, University of New South Wales, NSW 2052, Australia
†School of Information Technologies, University of Sydney, NSW 2006, Australia
{*tahirm,llibman*}*@cse.unsw.edu.au, hooman.dehkordi@sydney.edu.au, sanjay@cse.unsw.edu.au*

Abstract

There is growing interest in recent years in routing methods for wireless networks that leverage the broadcast nature of the wireless medium and the ability of nodes to overhear their neighbors' transmissions. Such methods include *opportunistic routing (OR)*, which generally choose the next hop on a routing path only after the outcome of the previous transmission is known; and wireless *network coding (NC)*, which linearly combines packets from different flows coexisting in the network. In this paper, we study the potential benefits of forwarding schemes that combine elements from *both* the OR and NC approaches, when traffic on a bidirectional unicast connection between two nodes is relayed by multiple common neighbors. We present a theoretically optimal scheme that provides a lower bound on the expected number of transmissions required to communicate a packet in both directions as a function of link error probabilities, and demonstrate that this bound can be up to 20% lower than with either OR or NC employed alone even in a small network. Using simulation, we further explore the control overhead in a direct implementation of the scheme with a simple coordination mechanism and show that the optimal bound can be closely approached for a wide range of link error rates.

Keywords: Opportunistic routing; network coding; wireless networks; optimal forwarding.

1. Introduction

The majority of routing protocols employed in multi-hop wireless networks are based on the traditional layered approach, using shortest-path and similar

A preliminary version of a part of this work was presented in [1].

graph theory-based algorithms which view every wireless channel between two nodes as a "link" in the same sense as in wired networks. However, in general, unless fine-tuned directional antennas are employed, wireless transmissions can be overheard by additional unintended nodes in the vicinity of the sender and receiver, causing interference to their own communications and resulting in additional constraints on links that may not be used simultaneously. This fact has led to considerable research into methods to overcome the interference and maximize network capacity, ranging from channel assignment methods based on coloring of the link conflict graph ([2] and references therein), to space-time scheduling methods such as STDMA [3].

In recent years, there has been growing recognition of the broadcast nature of the wireless medium as a virtue rather than a vice, with an increasing number of studies proposing ways to exploit the ability to overhear neighboring nodes' transmissions. Most of the related research has been in the physical layer, with considerable progress achieved on *cooperative relaying* methods to improve channel capacity in a single-hop setting (cf. [4] for a detailed overview). At higher layers, which are the focus of this paper, the proposed methods for leveraging the wireless overhearing capability can be broadly classified into two major categories: opportunistic routing and wireless network coding. In *opportunistic routing (OR)*, the next-hop forwarding node is chosen on-the-fly, rather than determined in advance by a routing protocol. This is exemplified by the ExOR protocol [5], where batches of packets are simply broadcast without acknowledgments; after every batch, each neighbor (in a predetermined priority order) forwards the packets it has successfully received, skipping those that it could overhear to be already forwarded by other neighbors. In *network coding (NC)*, data packets from different flows are mixed, thereby allowing a single transmission to hold different information content for different receivers. For example, the COPE protocol [6] illustrates how the number of transmissions is reduced in a bidirectional flow between a pair of endpoints via one or more intermediate relays, when the relays can broadcast bitwise-XOR combinations of messages from both sides (rather than forwarding each one separately), allowing each endpoint to recover the other's message by XOR-ing it again with its own.[1]

While both OR and NC approaches have a similar goal, namely to reduce transmissions by exploiting the free broadcasting feature in wireless networks, they are unrelated techniques that target different network conditions. Indeed, opportunistic routing is valuable in low-quality wireless environments, where channels have high error rates and volatility (e.g. due to fading). On the other hand, network coding achieves its best performance improvement with perfectly error-free channels, and is highly sensitive to errors. This naturally raises the question whether forwarding schemes that combine elements from

[1]The idea is not limited only to bidirectional flows between a pair of endpoints; generally, NC can be applied on packets from several flows intersecting at a common node, provided that packets from each flow can be overheard by all the next-hop nodes of all other flows.

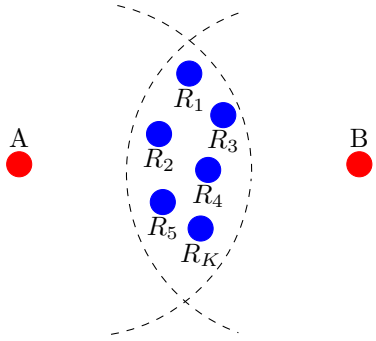


Figure 1: Bidirectional communication with multiple neighbors.

both techniques may achieve a further performance gain in practical networks, with "medium"-quality channel conditions that are between the above extremes. A typical scenario is shown in Figure 1, with two nodes (A,B) that engage in a bidirectional connection, where any of their multiple neighbors can potentially forward their packets. It is clear in this case that neither a pure OR method (e.g. ExOR), used independently in both directions without accounting for coding opportunities, nor a pure NC method (e.g. COPE) used with a single pre-selected neighbor, will achieve the best outcome if the link error rates are medium-to-high.

Our contribution in this paper is as follows. First, we demonstrate that, in general, naïve opportunistic network coding methods are sub-optimal unless the network is symmetrical (i.e. all neighbor nodes have identical link quality to both endpoints), and may even perform worse than NC over a single pre-chosen neighbor. Motivated by the above observation, we then formally define an *optimal* forwarding scheme as one that minimizes the expected total number of data transmissions required to deliver one packet in each direction of the flow. The key distinguishing features in the schemes we consider are that (1) the forwarding decision may be based on the full reception state of all nodes, not only the number of coding opportunities; and (2) we allow opportunistic forwarding to be applied *after* network coding, i.e. a node may retransmit a coded packet even if it is unable to decode its components by itself. We present a dynamic programming algorithm that finds the optimal scheme in a generic network instance with given link error rates, and demonstrate numerically that, even in a simple network with two intermediate relays, the expected number of transmissions can potentially be improved by a further 20% from existing approaches. Since the optimal scheme requires perfect knowledge of the set of packets received in every node at all times, it provides a lower bound on the average number of transmissions that can be achieved with any scheme in practice. Subsequently, we study the performance of an implementation of the optimal scheme with a simple coordination mechanism, where all nodes report the reception state in a series of acknowledgments after every data packet. We show via simulation that the performance of the scheme remains robust even under relatively high

loss rates of coordination messages (acknowledgments), and that most of the benefit of the theoretical optimal scheme can still be achieved even after the coordination overhead is taken into account.

The rest of this paper is organized as follows. After reviewing the related literature in Section 2, we present the system model and formally define optimal forwarding schemes in Section 3. We present the dynamic programming algorithm for finding the optimal scheme in Section 4, and illustrate the resulting performance numerically in Section 5. In Section 6, we consider the impact of the coordination overhead, and in particular of possible losses of coordination messages, on the resulting performance. Section 7 concludes the paper and discusses some directions for future extension of our results.

2. Related Work

The concept of opportunistic routing dates back to the Selection Diversity Forwarding (SDF) proposal [7], where each packet is broadcast with a list of potential next hop forwarders; neighbors that overhear the packet and find themselves on the list reply with acknowledgments that (optionally) contain their forward link state information, which is then used by the source to choose the best next hop. Some later proposals, such as link-layer anycast [8] and Geographic Random Forwarding (GeRaF) [9], are essentially variations on the same idea that use RTS/CTS exchanges, instead of actual data packets, to determine the next hop. Opportunistic routing methods became popular after the demonstration of the performance boosts achieved by the ExOR protocol [5], in which there is no explicit selection of the next hop at all; rather, packets are broadcast unacknowledged in batches, and after every batch, each neighbor (in a predetermined priority order) forwards the packets it has successfully received, skipping those that were already overheard being forwarded by other neighbors. An extension of the ExOR approach using *intra-flow* network coding (MORE), where the source node creates random linear combinations of packets within a batch and keeps sending them until receiving an acknowledgment for the entire batch from the destination, was proposed in [10]; subsequently, protocols such as CodeOR and PipelineOR [11, 12] have alleviated MORE's "stop-and-wait" nature between consecutive batches, to allow a sliding window among the batches or a pipeline of packets coded across different batches. The intra-flow coding of packets from a single source carries various practical benefits, such as being less sensitive to topology parameters and link characteristics, and allowing multiple cooperative nodes to operate in a distributed manner without explicit per-packet coordination. However, we point out that intra-flow coding in itself cannot improve the throughput of a unicast flow (or, equivalently, reduce the expected number of transmissions required to communicate a given amount of information) beyond what is achieved with opportunistic routing of individual packets, if the network topology parameters are known and full feedback about the outcome of each transmission is available [13].

On the other hand, protocols that use *inter-flow* wireless network coding, such as COPE proposed in [6], are based on broadcasting bitwise-XOR com-

binations of packets from different sources (e.g. from opposite endpoints of a bidirectional flow between the same endpoints), rather than forwarding each one separately; as a result COPE can improve the throughput even for unicast flows, since such combined packets carry different information for different receivers (unlike with intra-flow coding). The COPE protocol itself, notably, does not deal with routing of the flows, which is decoupled from the coding process. Some recent works on *coding-aware routing* protocols propose to adapt routing paths so as to attempt to increase the coding opportunities in a network; this was first motivated in [14], which introduced a wireless link metric based on expected number of coded transmissions (ECX). A more thorough discussion of the desirable properties of coding-aware routing metrics was provided in [15], together with a distributed mechanism for discovering potential coding opportunities on the available paths. Subsequently, the On-demand Coding-Aware Routing (OCAR) protocol described in [16] focused on the trade-off between maximizing the coding opportunities among flows (which requires routing paths to intersect in common nodes) and minimizing interference (which seeks to push routes away from each other). A detailed theoretical treatment of optimal coding-aware routing was given in [17], using a linear programming formulation to model the trade-off between coding opportunities and interference in the routing of multiple flows.

While all the above studies consider how to increase coding opportunities by improvements to *predetermined* routes, very few works have considered the application of (inter-flow) NC in the context of *opportunistic* routing. One notable example is the BEND proposal [18], which introduced a local heuristic approach for 'bending' routes opportunistically towards a common relay node within a two hop region. Our work goes beyond the limited idea of BEND, and allows OR and NC to be considered jointly and optimized among all possible transmissions at the same time; in other words, there is no *a priori* selection neither of routes nor the node(s) responsible for network coding. Rather, the choice of the packet (plain or coded) to transmit is made opportunistically based on the full reception state of all nodes. In particular, an important difference of our approach is that nodes are allowed to use opportunistic forwarding *after* network coding, i.e. it is possible for coded packets to be retransmitted in the network by nodes that are unable to decode them by themselves. This is different to the studies listed above, which only allowed coding opportunities to be considered where the combined packet would be able to be decoded immediately into the original native packets by each of the next hops on every flow.

3. Model and Problem Definition

We consider a wireless network with two nodes, A and B, that transmit a stream of fixed-size packets to each other. We assume that A and B are out of each other's direct transmission range; hence, they need their packets to be relayed in both directions by any of their K common neighbors (Figure 1). A neighbor node $R_k, k = 1, \ldots, K$ has a probability to successfully overhear a packet transmitted by A, B, or a peer neighbor R_j denoted by P_{A,R_k}, P_{B,R_k},

and P_{R_j,R_k} respectively. We assume these probabilities are independent among successive transmissions, i.e. that the link between any pair of nodes is memoryless at the packet time scale, which is generally the case if reception failures are predominantly caused by noise or fast fading. In addition, for simplicity of presentation, we will assume that these probabilities are independent among different links (i.e. node pairs), and that they are symmetrical in both directions of each link, i.e. P_{A,R_k}, P_{B,R_k} and P_{R_j,R_k} are also the probabilities for a packet transmitted by R_k to be received successfully at A, B, or R_j, respectively. Nevertheless, as will be explained in Section 4, our approach and results can be readily applied in networks with asymmetric and/or correlated links as well.

We emphasize that some of the overhearing probabilities may be set to zero (corresponding to pairs of nodes that cannot overhear one another at all). Thus, the model can represent a general network topology and is not limited to a one-hop or two-hop setting. For example, it is possible for the shortest path between A and B to consist of at least H hops, if the nodes are partitioned into $H+1$ groups ($0, \ldots, H$, with group 0 defined to consist of the source node A only, and group H of node B only), such that links with positive probability exist only between nodes within the same group or between nodes in adjacent groups. Since, as explained below, the focus of this work is on optimal forwarding schemes that minimize the expected number of transmissions, we assume (without loss of generality) that only one transmission can successfully take place anywhere in the network at any one time. Note that, in a small network where all the nodes are within a common range of interference, the optimal bound on the number of transmissions translates directly to an equivalent bound on the *latency* as well. For a larger network, where spatial reuse (i.e. concurrent transmissions) is possible, an optimal forwarding policy with respect to expected latency may not be equivalent to the one that minimizes the expected number of transmissions, and we leave the corresponding optimization variation for future work.

We define the *information state* at a node to be the set of packets it has received or overheard, and generically define a *forwarding scheme* as a mapping from the vector of information states at all nodes to a choice of node and packet for the next transmission (only one transmission is allowed at any time to avoid collisions). We are explicitly interested in forwarding schemes that allow network coding, i.e. transmissions of a coded combination of several packets in a node's information state.

The performance metric we use to evaluate and compare forwarding schemes is the expected number of data packet transmissions required to deliver one packet in each direction, i.e. one packet from A to B (henceforth called "packet a") and one packet from B to A ("packet b"). Note that this metric counts only *data* transmissions, and ignores the overhead of any feedback (such as acknowledgments) that would be required to synchronize the information states among nodes. Thus, we effectively assume that all nodes have full knowledge of their peers' information states after every packet transmission, and our subsequent analysis of the optimal forwarding scheme therefore provides a lower bound on the number of transmissions achievable in the ideal case of error-free feedback. From a practical standpoint, an implementation of the optimal scheme would

require a feedback mechanism that includes a series of acknowledgments from all nodes after every data transmission. A simple such mechanism is discussed later in Section 6, where the overhead of feedback exchanges is evaluated explicitly, including in the case where the feedback may not be perfect, i.e. acknowledgments can suffer from a nonzero loss rate as well.

To provide an initial insight and motivate the subsequent discussion of optimal schemes in a general network, we first demonstrate the limitations of some common forwarding schemes in a simple idealized example scenario, where all neighbors have identical link qualities to each endpoint and cannot overhear their peers; that is, $P_{A,R_k} = P_{AR}$, $P_{B,R_k} = P_{BR}$, and $P_{R_j,R_k} = 0$ for all $j, k = 1, \ldots, K$.

3.1. Traditional static routing

In traditional static routing, the complete path from source to destination is determined prior to the transmission. Since all relay nodes are equivalent, whichever one is chosen to be the intermediate forwarder, the expected number of transmissions required to communicate a packet in either direction between nodes A and B is $\frac{1}{P_{AR}} + \frac{1}{P_{BR}}$. Therefore, the total expected number of transmissions in both directions is

$$\frac{2}{P_{AR}} + \frac{2}{P_{BR}}. \tag{1}$$

3.2. Opportunistic routing (e.g. ExOR [5])

In opportunistic routing, the node to forward the packet to its destination is not selected in advance but, rather, chosen after the transmission by the source, and a retransmission by the source is required only if none of the potential relays copied the packet successfully.

Consider packet a first (from A to B). The probability of a successful transmission from A to at least one of the relay nodes is $1-(1-P_{AR})^K$. Thereafter, one of the relays retransmits the packet to its destination, which requires a further $\frac{1}{P_{BR}}$ transmissions. The calculation is similar for packet b in the opposite direction. Hence, the expected total number of transmissions required is

$$\frac{1}{1-(1-P_{AR})^K} + \frac{1}{1-(1-P_{BR})^K} + \frac{1}{P_{AR}} + \frac{1}{P_{BR}}. \tag{2}$$

3.3. Network coding (e.g. COPE [6])

In network coding, a single relay node is chosen to be the forwarder in both directions. However, compared to the traditional routing case, as long as neither packet has reached its destination, a transmission by the relay counts only once in both directions (as soon as one of the packets is successfully received, further relay transmissions are identical to the static routing case). Therefore, network coding uses on average $\frac{1}{1-(1-P_{AR})(1-P_{BR})}$ fewer transmissions than static routing; hence its expected number of transmissions is

$$\frac{2}{P_{AR}} + \frac{2}{P_{BR}} - \frac{1}{1-(1-P_{AR})(1-P_{BR})}. \tag{3}$$

3.4. Simple opportunistic network coding [18, 19]

Finally, we consider a simple combination of network coding with opportunistic routing, which in our simplified example scenario operates as follows. Each of the packets a, b is retransmitted as necessary by its respective source until received by at least one relay node. Subsequently, if there exists a relay node that successfully overheard both packets, network coding is performed; otherwise, the packets are forwarded by separate relays independently.

Once each packet has been received by at least one relay, a network coding opportunity is present if one of the relay nodes has received both packets a, b. To compute the conditional probability of this event, we observe that, for any specific relay, the *a priori* probability to receive both packets is $P_{AR} \cdot P_{BR}$; hence the *a priori* probability to have at least one relay receiving both packets is $1 - (1 - P_{AR}P_{BR})^K$. This must be divided by the *a priori* probability of the conditional event that is now known to have occured, i.e. that each of the two packets has been received by at least one relay. We therefore obtain that the probability of a network coding opportunity is

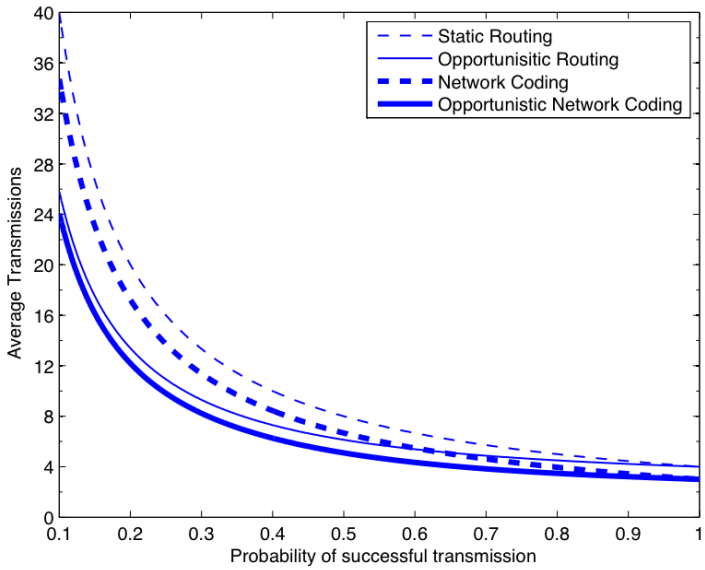
$$P_{nc} \triangleq \frac{1 - (1 - P_{AR}P_{BR})^K}{[1 - (1 - P_{AR})^K] \cdot [1 - (1 - P_{BR})^K]}. \tag{4}$$

Hence, we conclude that the expected total number of transmissions for this scheme is

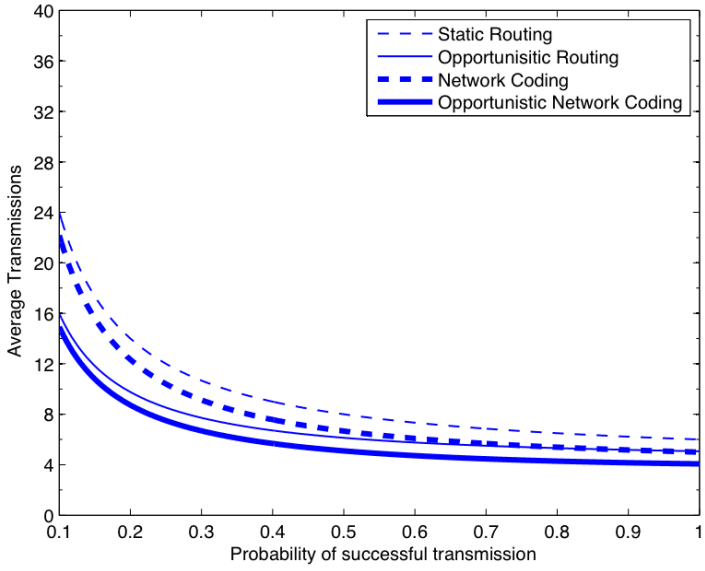
$$\frac{1}{1 - (1 - P_{AR})^K} + \frac{1}{1 - (1 - P_{BR})^K} + \frac{1}{P_{AR}} + \frac{1}{P_{BR}} - P_{nc}\frac{1}{1 - (1 - P_{AR})(1 - P_{BR})}. \tag{5}$$

Figure 2 demonstrates the values resulting from expressions (1)–(5) in a network of $K = 4$ neighbors. Specifically, Figure 2a shows the results when both P_{AR} and P_{BR} are varied between 0.1 and 1 together (maintaining $P_{AR} = P_{BR}$), whereas in Figure 2b, one of the probabilities is kept constant at $P_{AR} = 0.5$, and only P_{BR} is varied. We observe from Figure 2a that, when successful transmission probabilities are low, OR alone achieves a significant improvement over static routing, whereas NC performs nearly as badly as static routing. The reverse is true when these probabilities approach 1; then, NC considerably reduces the number of transmissions, whereas the ability to choose the forwarding relay opportunistically is of little use. The cross-over between the two curves occurs around $P_{AR} = P_{BR} \approx 0.624$. Importantly, we note that the opportunistic network coding (ONC) scheme is able to combine the improvements of each of its "components"; thus, it is most valuable (in percentage terms) with medium-range link probabilities (or, inversely, error rates) of around 0.5. The latter observation is further supported by Figure 2b, where setting $P_{AR} = 0.5$ emphasizes the improvement achieved by the ONC scheme in the entire range of P_{BR}.

However, it would be wrong to conclude from this example evaluation in symmetrical networks that the ONC scheme would perform well in the general case as well. To that end, consider the network in Figure 3 with $P_{A,R_1} = 0.5$, $P_{B,R_1} = 0.9$, $P_{A,R_2} = 0.9$, $P_{B,R_2} = 0.1$, and $P_{R_1,R_2} = 0$. Clearly, node R_1 is



(a) $0.1 \leq P_{AR} = P_{BR} \leq 1$



(b) $P_{AR} = 0.5, 0.1 \leq P_{BV} \leq 1$

Figure 2: Performance of common strategies: (a) for $0.1 \leq P_{AR} = P_{BR} \leq 1$; (b) for $P_{AR} = 0.5$, $0.1 \leq P_{BR} \leq 1$.

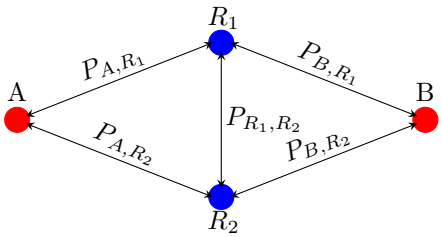


Figure 3: A generic network with two relay nodes.

the better relay overall, and simple NC using it as the sole forwarder achieves an expected number of transmissions of $\frac{2}{0.9} + \frac{2}{0.5} - \frac{1}{1-(1-0.9)(1-0.5)} \approx 5.17$. However, with ONC, a common outcome will be that packet a is only received by R_2, and packet b only by R_1, and they are forwarded individually to their respective destinations, incurring a high cost on the link $R_2 \leftrightarrow B$. It can be verified (by extending expressions (4)–(5) to the asymmetrical case; we omit the details for brevity) that the overall expected number of transmissions in the ONC scheme comes to approximately 8.857, i.e. much worse than network coding via R_1 alone. We will return to this example in Section 5, and show that, in fact, none of the schemes considered above achieves the optimal performance in this network.

4. Optimal Scheme Solution Algorithm

We proceed to describe the algorithm for finding the optimal forwarding scheme for the network model of Figure 1. First, we define the formal notation for the *network state* vector at any particular time instant, i.e. the $(K+2)$-tuple of the information states in all nodes: $\mathbf{S} = \langle S_A, S_B, S_{R_1}, \ldots, S_{R_K} \rangle$, where $S_A \in \{\{a\}, \{a,b\}\}$, $S_B \in \{\{b\}, \{a,b\}\}$, and $S_{R_k} \in \{\varnothing, \{a\}, \{b\}, \{a,b\}\}$, $k = 1, \ldots, K$. Here, a state of $\{a\}$ means that the respective node has so far overheard packet a but not packet b, and vice versa for $\{b\}$. We emphasize that, for a relay node R_k, $S_{R_k} = \{a,b\}$ may either mean that R_k overheard the two packets individually, or that it overheard their coded combination (e.g. their XOR). Even though in the latter case the node cannot decode the individual packets, it makes no difference to the node's actions in the optimal scheme, since it is never worse to transmit the coded combination than an individual packet anyway; this intuitive observation is proved formally later in Lemma 1.

Recall that a forwarding scheme is formally defined in our context as a mapping that, for every network state vector, specifies which node will make the next transmission and of which packet. Under a particular forwarding scheme, we define the *score* of a network state, denoted by $C(\mathbf{S})$, as the expected number of transmissions from that state until both endpoint nodes receive their packets. Thus, all network states with $S_A = \{a,b\}$ and $S_B = \{a,b\}$ have a score of 0; these are called *terminating states*. Note that some state vectors may not be able

to occur under particular forwarding schemes; for example, under static routing (or network coding) via R_1 in both directions, $S_A = \{a, b\}$ is not possible unless $b \in S_{R_1}$, and similarly $S_B = \{a, b\}$ requires $a \in S_{R_1}$.

We now derive the expressions describing the relationships among the scores of different states. First, suppose that for a particular state $\mathbf{S} = \langle S_A, S_B, S_{R_1}, \ldots, S_{R_K} \rangle$, the forwarding scheme selects node A to transmit its packet a. As a result, any relay node R_k that has not received the packet earlier will have a probability of P_{A,R_k} to overhear it, changing its own state from S_{R_k} to $S_{R_k} \cup \{a\}$. The score of $\mathbf{S}$ thus consists of 1 (counting this transmission of a) plus a weighted sum of the scores of all the possible new states to which the system can transition (including $\mathbf{S}$ itself if no new relays overhear the packet). Therefore,

$$C(\mathbf{S}) = 1 + \sum_{\mathcal{R} \in 2^{\left\{R_k | a \notin S_{R_k}\right\}}} \left[\prod_{R_k \in \mathcal{R}} P_{A,R_k} \prod_{R_k \notin \mathcal{R}} (1 - P_{A,R_k}) \right] \cdot C\left(\langle S_A, S_B, S_{R_k \notin \mathcal{R}}, S_{R_k \in \mathcal{R}} \cup \{a\} \rangle\right). \quad (6)$$

In (6), $\mathcal{R}$ is used to enumerate over all possible subsets of relay nodes which had not overheard the packet a previously. We make a slight abuse of notation and use $\langle S_A, S_B, S_{R_k \notin \mathcal{R}}, S_{R_k \in \mathcal{R}} \cup \{a\} \rangle$ to denote the new network state where $\{a\}$ is added only to the state of those nodes in the subset $\mathcal{R}$, while the state of all other nodes remains unchanged. The transition probability to this new network state then acts as the weight of its score in the sum.

We observe that one of the states in the sum in the right-hand side of (6) is $\mathbf{S}$ itself; this corresponds to $\mathcal{R} = \varnothing$, i.e. the case when no new relay node overhears the transmitted packet. Extracting it out of the sum, we get

$$C(\mathbf{S}) = \frac{1}{1 - P^A_{\mathbf{SS}}} + \sum_{\substack{\mathcal{R} \in 2^{\left\{R_k | a \notin S_{R_k}\right\}} \\ \mathcal{R} \neq \varnothing}} \frac{\prod_{R_k \in \mathcal{R}} P_{A,R_k} \prod_{R_k \notin \mathcal{R}} (1 - P_{A,R_k})}{1 - P^A_{\mathbf{SS}}} \cdot C\left(\langle S_A, S_B, S_{R_k \notin \mathcal{R}}, S_{R_k \in \mathcal{R}} \cup \{a\} \rangle\right), \quad (7)$$

where $P^A_{\mathbf{SS}} \triangleq \prod_{R_k | a \notin S_{R_k}} (1 - P_{A,R_k})$ is the probability of the network to remain in state $\mathbf{S}$ after a transmission by A. Expression (7) can be intuitively interpreted as follows: the score of $\mathbf{S}$ is the expected number of transmissions until a successful transition to another state occurs, followed by a weighted sum of state scores, where the weights this time are the respective *conditional* transition probabilities given that a transition to a new state has occured.[2]

[2]Note that the transition between (6) and (7) assumes that $P^A_{\mathbf{SS}} \neq 1$; this is always a safe assumption, since no forwarding scheme would logically choose a transmission by node A if there are no relay nodes with a positive link probability to A that have not already heard its packet.

For the case where the forwarding scheme selects node B to transmit the packet b, we similarly obtain

$$C(\mathbf{S}) = \frac{1}{1-P_{\mathbf{SS}}^{B}} + \sum_{\substack{\mathcal{R}\in 2^{\left\{R_k | b\notin S_{R_k}\right\}} \\ \mathcal{R}\neq\varnothing}} \frac{\prod_{R_k\in\mathcal{R}} P_{B,R_k} \prod_{R_k\notin\mathcal{R}}(1-P_{B,R_k})}{1-P_{\mathbf{SS}}^{B}} \cdot C\left(\langle S_A, S_B, S_{R_k\notin\mathcal{R}}, S_{R_k\in\mathcal{R}} \cup \{b\}\rangle\right), \quad (8)$$

where $P_{\mathbf{SS}}^{B} \triangleq \prod_{R_k | b\notin S_{R_k}} (1-P_{B,R_k})$.

We turn to consider the score of a state $\mathbf{S}$ where the forwarding scheme calls for a transmission of packet a by a relay node R_j. We assume that such a transmission makes sense, namely, R_j indeed has a copy of the packet and the destination B has not received it yet. Accordingly, we obtain the expression for the score in this case:[3]

$$C(\mathbf{S}) = \frac{1}{1-P_{\mathbf{SS}}^{R_j,a}} + \sum_{\substack{\mathcal{R}\in 2^{\{B\}\cup\left\{R_k | a\notin S_{R_k}\right\}} \\ \mathcal{R}\neq\varnothing}} \frac{\prod_{X\in\mathcal{R}} P_{X,R_j} \prod_{X\notin\mathcal{R}}(1-P_{X,R_j})}{1-P_{\mathbf{SS}}^{R_j,a}} \cdot C\left(\langle S_A, \begin{smallmatrix} S_B\cup\{a\} & \text{if } B\in\mathcal{R} \\ S_B & \text{if } B\notin\mathcal{R} \end{smallmatrix}, S_{R_k\notin\mathcal{R}}, S_{R_k\in\mathcal{R}} \cup \{a\}\rangle\right), \quad (9)$$

where $P_{\mathbf{SS}}^{R_j,a} \triangleq (1-P_{B,R_j}) \prod_{R_k | a\notin S_{R_k}} (1-P_{R_j,R_k})$.

A similar expression defines the score of a state $\mathbf{S}$ in which a node R_j transmits the packet b, where $b \notin S_A$:

$$C(\mathbf{S}) = \frac{1}{1-P_{\mathbf{SS}}^{R_j,b}} + \sum_{\substack{\mathcal{R}\in 2^{\{A\}\cup\left\{R_k | b\notin S_{R_k}\right\}} \\ \mathcal{R}\neq\varnothing}} \frac{\prod_{X\in\mathcal{R}} P_{X,R_j} \prod_{X\notin\mathcal{R}}(1-P_{X,R_j})}{1-P_{\mathbf{SS}}^{R_j,b}} \cdot C\left(\langle \begin{smallmatrix} S_A\cup\{b\} & \text{if } A\in\mathcal{R} \\ S_A & \text{if } A\notin\mathcal{R} \end{smallmatrix}, S_B, S_{R_k\notin\mathcal{R}}, S_{R_k\in\mathcal{R}} \cup \{a\}\rangle\right), \quad (10)$$

where $P_{\mathbf{SS}}^{R_j,b} \triangleq (1-P_{A,R_j}) \prod_{R_k | b\notin S_{R_k}} (1-P_{R_j,R_k})$.

Finally, we consider the expression for the score of $\mathbf{S}$ where a node R_j with $S_{R_j} = \{a,b\}$ transmits the coded combination of a and b. Following the same reasoning as before, we obtain

$$C(\mathbf{S}) = \frac{1}{1-P_{\mathbf{SS}}^{R_j,ab}} + \sum_{\substack{\mathcal{R}\in 2^{\{X | \{a,b\}\not\subseteq S_X\}} \\ \mathcal{R}\neq\varnothing}} \frac{\prod_{X\in\mathcal{R}} P_{X,R_j} \prod_{X\notin\mathcal{R}}(1-P_{X,R_j})}{1-P_{\mathbf{SS}}^{R_j,ab}} \cdot C\left(\langle \begin{smallmatrix} \{a,b\} & \text{if } A\in\mathcal{R} \\ S_A & \text{if } A\notin\mathcal{R} \end{smallmatrix}, \begin{smallmatrix} \{a,b\} & \text{if } B\in\mathcal{R} \\ S_B & \text{if } B\notin\mathcal{R} \end{smallmatrix}, S_{R_k\notin\mathcal{R}}, S_{R_k\in\mathcal{R}} \cup \{a,b\}\rangle\right), \quad (11)$$

[3]For conciseness and to avoid considering the link between R_j and B separately, we use the letter 'X' in (9) to denote any node in the set $\mathcal{R}$, which can be either B or one of the relay nodes R_k.

where $P_{\mathbf{SS}}^{R_j,ab} \triangleq \prod_{X|\{a,b\}\not\subseteq S_X}(1-P_{X,R_j})$.

We now focus more closely on the relationships among the states in the above expressions. Consider a graph G where the vertices are the 4^{K+1} possible network state vectors, and there is a (directed) edge from $\mathbf{S}$ to $\mathbf{S}'$ if and only if $\mathbf{S}'$ appears in one of the score formulas of $\mathbf{S}$, i.e. in the right-hand side of one of the expressions (7)–(11). Thus, every edge in the graph corresponds to a transmission of either a, b, or their coded combination, and a specific subset $\mathcal{R}$ of the new nodes receiving that transmission. Then G (the *state dependency graph*) is a DAG (directed acyclic graph). Indeed, a directed cycle is not possible in G, since for every directed link $\mathbf{S} \to \mathbf{S}'$, $\mathbf{S}'$ is a strict superset of $\mathbf{S}$; that is, $S_A \subseteq S'_A$, $S_B \subseteq S'_B$, $S_{R_k} \subseteq S'_{R_k}, k = 1, \ldots, K$, and $\mathbf{S} \neq \mathbf{S}'$.

Since the graph of dependencies among states is acyclic, the scores of all states in the optimal forwarding scheme can be computed by the following algorithm:

1. Initialize $\mathcal{T}$ to the set of all terminating states (i.e. where $S_A = S_B = \{a, b\}$); set $C(\mathbf{S}_\mathcal{T}) = 0$ for all $\mathbf{S}_\mathcal{T} \in \mathcal{T}$;
2. Choose a state $\mathbf{S}$ such that $\mathbf{S} \notin \mathcal{T}$ and all the directed edges from $\mathbf{S}$ in G are to states in $\mathcal{T}$ (such a state is guaranteed to exist due to G being cycle-free), and set $\mathcal{T} = \mathcal{T} \cup \{\mathbf{S}\}$;
3. Set $C(\mathbf{S})$ to the minimum of expressions (7),(8), and (9),(10),(11) for every $R_j, j = 1, \ldots, K$, and let OFS($\mathbf{S}$) record the corresponding transmission choice;
4. If every possible network state is in $\mathcal{T}$, stop; else goto step 2.

The correctness of the above algorithm follows from the fact that in every iteration, it finds the optimal action that leads to the lowest score for one state $\mathbf{S}$, provided that the optimal scores have already been found for all the states it depends on; therefore, by induction, it eventually finds the optimal score for the starting state $\langle\{a\}, \{b\}, \varnothing, \ldots, \varnothing\rangle$, while recording the optimal action in each state $\mathbf{S}$ in OFS($\mathbf{S}$). In other words, this is a dynamic programming algorithm, which resembles the Bellman-Ford algorithm for shortest paths in graphs or the Viterbi algorithm for finding a maximum-likelihood sequence of states [20]. The running time complexity of the algorithm is found by noting that it executes one iteration for each network state, i.e. 4^{K+1} iterations in total, with up to $3K + 2$ score evaluations in each iteration. This complexity is usually reasonable since K, the number of common neighbors of two nodes in a wireless network, is typically small in practice.

Remark. For simplicity of presentation, the above score expressions (7)–(11) were derived for the case where the probabilities of successful reception/overhearing between each pair of nodes are independent and symmetrical. We observe that the score expressions can be extended in a straightforward manner to the case where these probabilities are asymmetrical and/or have a non-negligible level of spatial correlation [21]; to that end, the score of each state should still be based on the weighted sum of the scores of all its dependent states, using the actual transition probabilities among the states (which need to account for the spatial correlations and may no longer involve simple products of reception probabilities

on different links). The correctness of the dynamic programming algorithm can be shown to hold in the extended case using the same considerations as above. The only essential assumption is that the transition probabilities are memoryless (namely, independent over time); this is required so that the score of each state can be well-defined, independently of the history of the system prior to reaching that state. The necessary extensions to consider optimal forwarding schemes for networks with time-correlated links (on a time scale of a full packet and beyond) are left for future work.

Before concluding this section, we now formally prove the intuitive property that, if a node R_j has both packets a and b, it is never worse to transmit the coded combination of both packets than just one of them. This provides the necessary justification to the claim that the state of a node that either overheard both packets individually or just their coded combination can equally be represented by $\{a, b\}$.

Lemma 1. *There exists an optimal forwarding scheme that does not involve a transmission of an individual packet (a or b) by a node R_j where $S_{R_j} = \{a, b\}$.*

Proof. Consider an optimal forwarding scheme for which, in a state $\mathbf{S}$ such that $S_{R_j} = \{a, b\}$, the algorithm chooses node R_j to transmit packet a in step 3 (the proof is identical if it transmits packet b instead). Assume, without loss of generality, that $\mathbf{S}$ is the first among all such states in the course of execution of the algorithm.

Consider an arbitrary directed path in G from the state $\mathbf{S}$ to some terminating state, which starts with a directed edge corresponding to the transmission of packet a by R_j (for some $\mathcal{R} = \mathcal{R}_{\mathbf{S}}$). Then, there exists an alternative path which starts from $\mathbf{S}$ along the edge corresponding to the transmission of the combination of $\{a, b\}$ instead (for the same $\mathcal{R}_{\mathbf{S}}$), and thereafter follows the sequence of edges that corresponds to the same transmitted packets and subsets $\mathcal{R}$ as the edges of the former path. In other words, both paths represent the same sequence of transmissions (apart from the first one) and the same pattern of success outcomes on all links in the actual network; hence, they have the same sequence of transition probabilities. Since the alternative path traverses network states that differ from the network states in the former path only by the fact that the nodes in the subset $\mathcal{R}_{\mathbf{S}}$ have a node state of $\{a, b\}$ instead of $\{a\}$, it necessarily leads to a terminating state as well (in fact, it may reach a terminating state sooner than the former path). It follows that the scores of the states along the latter path are not larger than those of the corresponding states along the former path. Consequently, the score of $\mathbf{S}$ with a combined transmission of $\{a, b\}$ (i.e. calculated by expression (11)) is not larger than with a transmission of a alone (i.e. calculated by (9)), and the algorithm can therefore choose the combined transmission in step 3 instead. □

5. Numerical Evaluation

In this section, we demonstrate the optimal forwarding scheme and compare its performance with the other aforementioned schemes for a range of example

scenarios. This comparison has already commenced in Section 3, for the simple symmetrical case where all the relay nodes are equivalent ($P_{A,R_k} = P_{AR}$ and $P_{B,R_k} = P_{BR}$ for all k), and unable to overhear each other's transmissions ($P_{R_j,R_k} = 0$ for all j, k). It turns out that, in this case, the ONC scheme (i.e. the last of the four schemes discussed in Section 3) is in fact the optimal one. Indeed, since all the neighbor nodes are equivalent, once each of the packets a and b has been received by a neighbor, there can be no gain in retransmitting any of the packets by the original source in an attempt to copy it to another neighbor as well. Thus, the ONC scheme, which performs network coding whenever an opportunity exists or reverts to simple opportunistic forwarding otherwise, achieves the lowest possible expected number of transmissions.

In a symmetrical scenario where all neighbor nodes are equivalent but can also overhear each other with a positive probability (i.e. $P_{R_j,R_k} = P_{RR} > 0$ for all $j, k = 1, \ldots, K$), the performance of the optimal scheme improves further somewhat. This can be explained as follows. Even if there is no coding opportunity initially and the scheme proceeds to forward an individual packet, that packet can then be overheard by another relay holding a copy of the other packet, thus creating a coding opportunity and reducing the expected number of remaining transmissions from that point. Figure 4 demonstrates how the performance of the optimal scheme depends on P_{RR} for several combinations of P_{AR} and P_{BR} in a symmetrical network with $K = 2$. We observe that the improvement is not very significant, and is less than about 5% between $P_{RR} = 0$ and $P_{RR} = 1$. Incidentally, we point out that in order to maximize the expected saving of transmissions by the aforementioned possibility of a belated coding opportunity created by overhearing among relays, the packet with the *worse* link to the destination (e.g. packet a if $P_{AR} > P_{BR}$, and packet b if $P_{AR} < P_{BR}$) should be the one going first in the opportunistic forwarding phase.

We now turn to discuss general, asymmetrical scenarios where relay nodes are no longer equivalent to each other. We begin by considering again the example from the last paragraph of Section 3. In that example, based on the network in Figure 3 with $P_{A,R_1} = 0.5$, $P_{A,R_2} = 0.9$, $P_{B,R_1} = 0.9$, $P_{B,R_2} = 0.1$, $P_{R_1,R_2} = 0$, it was shown that the opportunistic network coding scheme was in fact worse than network coding over a static route using R_1 only. We now describe the optimal scheme found by the dynamic programming algorithm in this instance.

1. Transmit packet a by node A and packet b by node B until each of the packets is received by a relay;
2. While none of the endpoints received the opposite packet:
   - if R_1 has both a and b, let R_1 transmit their combination;
   - else, if R_2 has both a and b *and* R_1 does not have a copy of packet b, let R_2 transmit their combination;
   - else, if R_1 has b only (thus R_2 has a or both packets), retransmit packet a by node A;
   - else, i.e. if R_1 has a only and R_2 has b only, forward the packets individually;

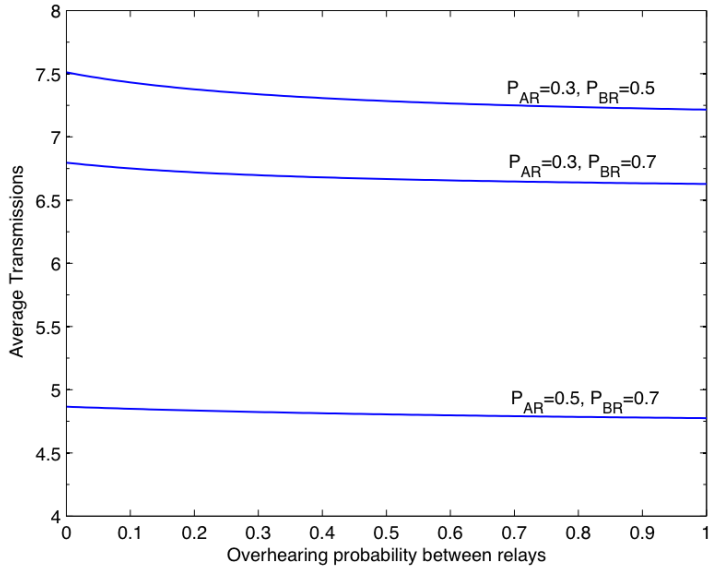
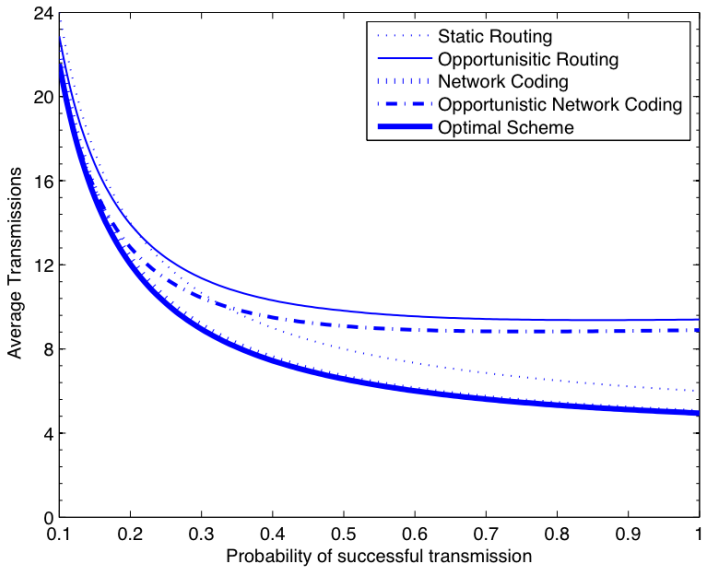


Figure 4: Optimal scheme performance in symmetrical scenario ($K = 2$), as a function of P_{RR}.

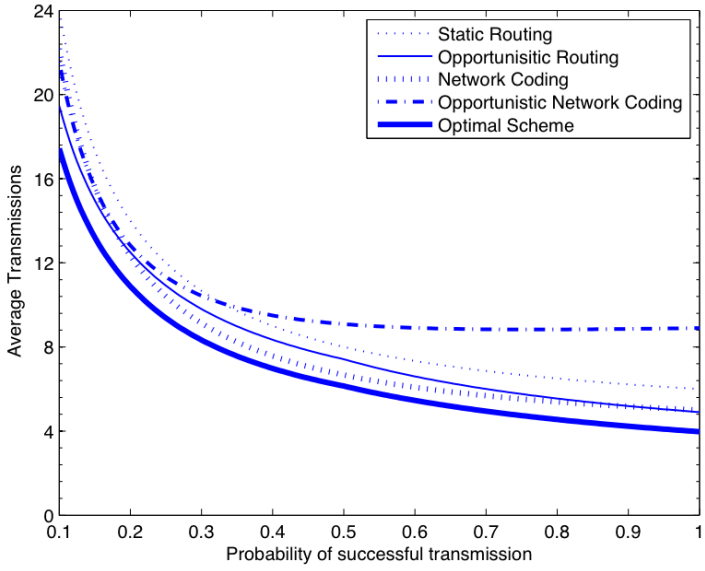
3. If A receives packet b first, use fixed routing via R_1 for packet a thereafter (i.e. ignore R_2 even if it has a, and retransmit the packet from A if necessary);
4. If B receives packet a first, use opportunistic routing via either relay for packet b (with a preference for R_2 if both have it at the same time).

We observe that the optimal scheme is different from simple opportunistic network coding, as the decision to perform network coding by node R_2 (the lower-quality relay) depends on the state of node R_1: the coding should go ahead if R_1 has no packets or only overheard packet a, whereas if R_1 already has a copy of b, it is preferable to retransmit packet a from A, insisting on the network coding to be done by R_1 thereafter. Also, note that the network coding by R_2 is only done until the combined transmission is received by A (for the small extra chance that node B may receive it as well); once A is in possession of packet b, R_2 is no longer used to deliver packet a to B. This scheme achieves an expected total number of transmissions of 5.115, which is slightly lower than 5.17 achieved by network coding over the static route via R_1.

To extend this example further, Figure 5a compares the performance of the various schemes for the two-relay network of Figure 3 where $P_{A,R_1} = 0.5$, $P_{B,R_2} = 0.1$, $P_{R_1,R_2} = 0$, and $P_{A,R_2} = P_{B,R_1}$ are varied together (kept equal) between 0.1 and 1. We observe that, throughout the entire range, the optimal scheme maintains a slight lead of about 1%–2% over network coding with a static path. The situation becomes more interesting in Figure 5b, where the



(a) $0.1 \leq P_{A,R_2} = P_{B,R_1} \leq 1, P_{R_1,R_2} = 0$



(b) $0.1 \leq P_{A,R_2} = P_{B,R_1} \leq 1, P_{R_1,R_2} = 1$

Figure 5: Performance comparison for $K = 2$ with $P_{A,R_1} = 0.5$, $P_{B,R_2} = 0.1$, and (a) $P_{R_1,R_2} = 0$; (b) $P_{R_1,R_2} = 1$.

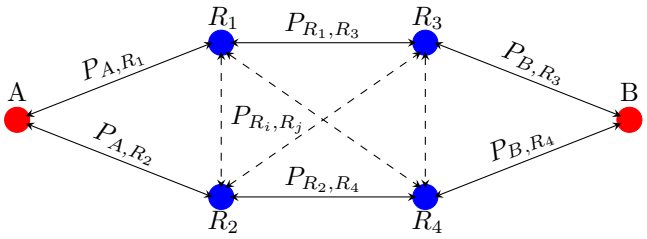


Figure 6: A multi-hop network with four relay nodes.

same range of scenarios is considered except that $P_{R_1,R_2} = 1$. Allowing the neighbors to overhear each other perfectly does not make a difference for the static routing and NC schemes, since the single best path in the network is still A $\leftrightarrow$ R_1 $\leftrightarrow$ B. Neither does it impact the performance of the ONC scheme, or for that matter any protocol where a node with the first coding opportunity keeps transmitting the combined packet until success, and does not opportunistically route it further via other nodes (e.g. BEND [18]). On the other hand, the performance of the optimal scheme (and of the pure OR scheme) improves markedly, since a transmission by R_2 of packet a (or the coded combination of a and b) is no longer 'wasted' when the link $R_2 \leftrightarrow$ B fails; instead, it is overheard by R_1 which can then take over and deliver the packet to its destination with fewer transmissions. Indeed, the optimal scheme here no longer favors a retransmission of packet a by its source A rather than by node R_2, if the latter has it. With respect to the original example (namely, $P_{A,R_2} = P_{B,R_1} = 0.9$), the expected total number of transmissions is reduced to 4.23, almost 20% lower than the next-best option (which is still network coding via R_1). This is in stark contrast to the improvement of only a few percent in the *symmetrical* scenarios, as shown in Figure 4.

In order to demonstrate how the optimal forwarding scheme performs in the case of a multi-hop network, we further consider the network depicted in Figure 6, which is a 3-hop analog of the 2-relay network of Figure 3 with each of the original relays split into two nodes with a good link between them, but no longer able to directly overhear or be overheard by one of the endpoints. Consider the overhearing probability values of $P_{A,R_1} = P_{B,R_4} = 0.5$, $P_{A,R_2} = P_{B,R_3} = 0.9$, and $P_{R_1,R_3} = P_{R_2,R_4} = 1$, with overhearing probabilities on all other inter-relay links (except P_{R_1,R_3} and P_{R_2,R_4}) set to 0; in other words, overhearing is not possible over the links represented by dashed lines in Figure 6, and the topology again consists of two separate paths between the endpoints. It can be easily verified that, for this case, the best among the original schemes of Section 3 is network coding over either of the two static paths, achieving an expected number of transmissions of $\frac{2}{0.5} + \frac{2}{0.9} + 1 \approx 7.222$ (i.e. one fewer transmission than simple independent forwarding in both directions, with the network-coded transmission initiated by either one of the relays along the path). This is considerably higher than the optimal bound of 6.64 transmissions achieved by the optimal forwarding scheme found by the algorithm of section 4, which is able to efficiently utilize

both paths opportunistically. In the other case where the overhearing probability among all the relay nodes is set to 1, the optimal number of transmissions drops further to approximately 5.329, which is still ahead of the second-best option of network coding over the optimal path of A$\leftrightarrow$ R_2 $\leftrightarrow$ R_3 $\leftrightarrow$B with a score of $\frac{4}{0.9} + 1 \approx 5.444$.

6. The Impact of State Synchronization Overhead

So far, we have considered the theoretical bound on the expected number of transmissions that can be achieved by optimal forwarding, without accounting for the overhead cost of synchronizing the network state among the participating nodes after every data packet. In this section we turn our attention to the synchronization overhead explicitly, and consider the benefit that can be still achieved by the optimal forwarding scheme even after this overhead is taken into account. We explain the considerations in the design of a robust feedback mechanism, and calculate the additional time taken by feedback messages under the assumption that the feedback channel is perfect, showing that the additional overhead is small in comparison to the typical savings from the reduced data packet transmissions. Subsequently, we use an OMNET implementation of the feedback mechanism to show by simulation how the optimal scheme performance is impacted by a nonzero loss rate of the feedback messages themselves, using both the 2-hop and 3-hop topology examples from the previous section.

6.1. Distributed synchronization mechanism

Given that every transmission decision in the optimal scheme depends on the latest information state in all nodes, a distributed implementation of the scheme requires the full network state to be synchronized in all nodes, so that each node can decide individually whether it should be the next one to transmit. Clearly, the network state synchronization process requires every node to send some sort of a feedback message or acknowledgment (ACK). However, a traditional simple ACK, which is only broadcast when a packet is received and only contains information about the reception in one node, is not well-suited for a distributed state synchronization, for the following reasons. First, it requires all nodes to be able to overhear each other's ACKs; in particular, this rules out any scenarios where the source and destination are out of direct range of each other. Second, every failure of any node to hear an ACK message will immediately lead to a loss of synchronization of the network state among nodes, potentially resulting in deadlock or collision due to conflicting decisions by the different nodes.

Accordingly, we henceforth consider *extended* feedback messages, where each ACK advertises the entire network state vector as perceived by the node (Figure 7). Whenever any node overhears an ACK, it updates its own view of the network state by noting any new information not already known about packets received or overheard in other nodes, even if these refer to nodes other than the one transmitting the ACK itself. The use of such extended ACKs allows state synchronization to be achieved even when the source and destination (or any

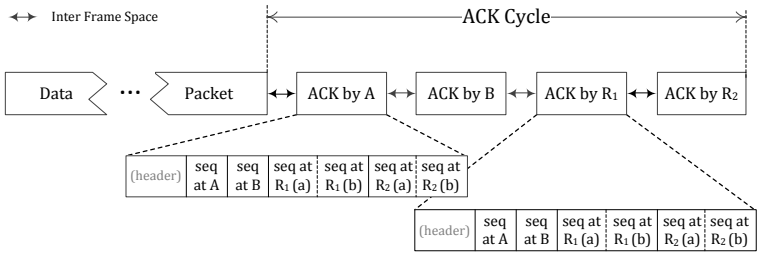


Figure 7: Structure and timing of the ACK messages for network state synchronization.

two nodes, for that matter) cannot directly overhear each other's ACKs, as the relevant information will be present in the ACKs of other relay nodes as well. In the same way, extended ACKs provide additional redundancy that increases the tolerance against random ACK losses as well; therefore it becomes less essential to have very low ACK loss rates in the network. As will become evident from the subsequent simulation results, the extended feedback mechanism maintains a robust performance even as ACK loss rates increase, for the price of only a mildly higher overhead cost.

We now proceed to estimate the overhead cost of the feedback that follows the transmission of a data packet. For this purpose, we assume that after a data packet is received, all nodes send out their ACK messages in a pre-defined fixed order that is known to all the nodes; this complete set of ACK transmissions following every data packet is referred to as an *ACK cycle*. Thus, each node can tell in advance how long it waits before transmitting its ACK, and will wait accordingly even if some other ACKs earlier in the cycle are missing, e.g. when the corresponding node(s) did not receive the packet. However, while waiting, each node should instantly update its view of the network state upon receiving any peer ACKs, and reflect it in its own ACK subsequently. Since the ACK order is fixed regardless of the identity of the transmitter, the ACK cycle duration consists of $K + 2$ ACK transmissions. In particular, all $K + 2$ nodes are assumed to take part in the ACK cycle, including the transmitter of the just-completed data packet (this boosts the redundancy in the dissemination of the network state information). The fixed ACK order that we use below is $A, B, R_1, R_2, \ldots, R_K$; note that, by putting the ACK messages from the endpoints first, notification about the reception of a packet at a destination can reach the opposite end within the same ACK cycle. The ACK cycle timing, as well as the content of each ACK message, are illustrated in Figure 7. Note that each ACK contains a total of $2K + 2$ state fields (seq), each representing a sequence number of the last packet received in each of the nodes (1 field for each of A and B, and 2 fields, i.e. one for each direction, for each of the relay nodes).

Assuming that a small time interval must be maintained between successive

ACK transmissions (such as the Short Interframe Space, $SIFS$, in IEEE 802.11) to allow for propagation delay differences, the total overhead of the ACK cycle is:

$$(K+2)\cdot\left[\frac{header+(2K+2)\cdot seq}{rate}+SIFS\right] \tag{12}$$

where *header* is the size of the fixed header of ACK packets, *seq* is the size of the sequence number (or equivalent field) containing the identity of the last packet received by a node, and *rate* is the raw bitrate of the ACK transmission. Using typical values from the IEEE 802.11b/g protocol as an example and substituting $header = 14$ bytes, $seq = 1$ byte, $rate = 1$ Mbps, and $SIFS = 10$ μsec, we get the total duration of the ACK cycle to be approximately 0.6 msec for $R = 2$ relay nodes, or 1 msec for $R = 4$ relay nodes. This can be compared to the transmission time of a maximum-size data packet (approximately 1.5 KB) at the same bitrate, which is in excess of 12 msec. Since the (rate-independent) short interframe spaces take a negligible fraction of an ACK cycle, the ratio between the duration of a data packet and an ACK cycle remains similar at higher bitrates as well. It can be concluded that a forwarding strategy requiring a full state synchronization after every packet is still worthwhile despite the additional overhead if it reduces the expected number of retransmissions by roughly 0.1 or more.

The above estimation was based on the assumption that all feedback messages are heard perfectly; then, the time spent in ACK cycles is indeed the only overhead cost of the mechanism. When the feedback channels are not perfect and the ACK messages can get lost or corrupted, inconsistent views of the network state may arise in different nodes. This inconsistency can lead to one of the following undesirable situations: either (1) *collision*, when two or more nodes decide to transmit the packet at the same time by applying the optimal scheme on different perceived network states; or (2) *deadlock*, when every node decides that some other node will transmit the packet, and an actual transmission never happens. In order to evaluate the impact of feedback losses and state inconsistencies on the resulting performance, we implemented the extended feedback mechanism, as described above, in the OMNET simulator platform.[4] Our implementation uses the following methods to resolve collision and deadlock situations. First, upon receiving a corrupted packet, a node will wait a duration of a full ACK cycle. During that time, if any ACKs are successfully received (indicating a successful reception of a data packet in some other nodes), the node immediately updates its view of the network state and joins the ongoing ACK cycle in a normal fashion. Otherwise, if nothing further happens on the channel after the corrupted reception for a full length of an ACK cycle, the node assumes that a collision has likely occured and that all other nodes have reached the same conclusion; an ACK cycle is then started

[4]The full source code of our implementation may be obtained from `http://www.cse.unsw.edu.au/~llibman/papers/OR_NC_implementation.rar`.

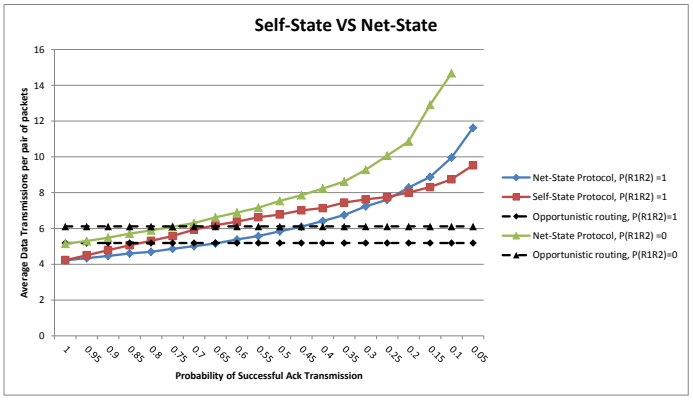
from that point, with the node taking its normal fixed position in the cycle.[5] Second, if either of the source nodes (A and B) receives neither a packet nor an ACK for the duration of a timeout, set to be equal to the transmission time of a maximum-size packet plus an ACK cycle, it initiates a transmission of an ACK on its own, thereby triggering relay nodes to join in an impromptu ACK cycle. This rule ensures that a deadlock is eventually resolved, by continuing to transmit ACKs periodically until the node(s) with the incorrect network state view eventually receive the correct state update.

6.2. Simulation results

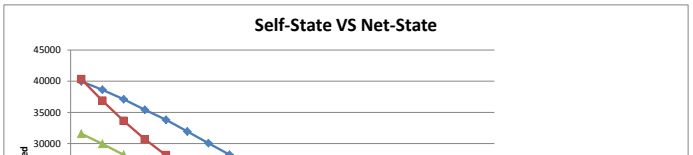
We simulate the performance of the optimal forwarding scheme, as described in Section 5, using again the example based on the topology of Figure 3 for which the expected number of transmissions per pair of packets in both directions was found above to be 5.115 when $P_{R1,R2} = 0$ and 4.23 when $P_{R1,R2} = 1$ (that was assuming the nodes are always synchronized with no inconsistencies). **Our aim is to explore how the expected number of transmissions per pair of packets is affected when the acknowledgment loss rate increases, due to synchronization breakdowns leading to an increasing incidence of deadlocks, collisions, or simply suboptimal transmission decisions. Note that, generally, the loss rate of acknowledgments need not be related to that of the data packets — in fact, the feedback channel is typically considered in practice to be more reliable than the data channel, since acknowledgments are short and therefore less prone to outages and bit errors than data packets. For this reason, in the following discussion, we control the loss rate of acknowledgments independently of that of the data packets.** The results are shown in Figure 8a, where the leftmost point on the curves corresponds to perfect feedback channels, and the reception probability of the ACK decreases (i.e. the loss rate increases) from there; for simplicity and due to space constraints, we only present the results for the case where the ACK probability is identical in all the links.

The three curves shown in Figure 8a correspond to the extended feedback mechanism (branded "net-state") with $P_{R1,R2} = 0$ and with $P_{R1,R2} = 1$, and, for comparison, the simple feedback where the ACK message from each node only notifies about its own state of reception, rather than the full state vector (referred to as "self-state" in the figure). Recall that the "self-state" mechanism cannot adequately be used for synchronization unless every node is able to hear every other node's ACK messages; for this reason, we do not evaluate it for $P_{R1,R2} = 0$, and even for the $P_{R1,R2} = 1$ case we artificially allow the endpoints A and B to hear each other's ACKs with the same probability as in all other links (whereas the simulation of the "net-state" mechanism is run with no direct link between A and B at all). Despite this artificial advantage in the simulation,

[5] The above rule only applies if the corrupted reception occurs while the node is idle and expecting to receive a data packet, and not if it is already in the midst of an ACK cycle; then, a corrupted reception is simply treated as a lost ACK and ignored.


Self-State VS Net-State
Average Data Transmissions per pair of packets
Probability of Successful Ack Transmission
Net-State Protocol, P(R1R2) =1
Self-State Protocol, P(R1R2) =1
Opportunistic routing, P(R1R2)=1
Net-State Protocol, P(R1R2) =0
Opportunistic routing, P(R1R2)=0

(a)


Self-State VS Net-State

it can be seen that the "self-state" mechanism is significantly inferior to the "net-state" one and much less robust as the ACK loss rate increases, for a range of reasonable ACK loss rate values. It is only when the ACK reception probability becomes very low (about 0.2 or less) that the "self-state" curve drops to become the lowest; again, we emphasize that this is attributed entirely to the above artificial advantage introduced in the simulation, and does not imply that "self-state" feedback is superior in any way with high ACK loss rates.

The dashed horizontal lines in Figure 8a show the average number of transmissions corresponding to using opportunistic routing of packets in both directions independently. This performance level represents the upper bound of what can be ideally achieved with distributed intra-flow coding techniques, such as MORE [10], which do not require state coordination or feedback among the participating nodes (and, consequently, do not depend on the ACK reception probability).[6] **As these figures demonstrate, when the loss rate on the feedback links is high and explicit state synchronization among the nodes becomes unreliable, it may be better in practice to employ a forwarding strategy that is suboptimal in principle, but does not need reliable feedback links for its implementation.**

Figure 8b presents the overall throughput of the optimal forwarding scheme (shown as the total number of packets delivered during the simulation, running for 500 sec), again as a function of the ACK reception probability (identical in all links) in the same three scenarios. This figure is arguably more representative of the performance, since it takes into account the additional time wasted in the recovery from deadlocks and collisions, as opposed to merely counting the number of transmissions. We observe that the performance degrades gracefully and roughly linearly with increasing ACK loss rate. **For the sake of comparison, the dashed horizontal line in Figure 8b shows the corresponding value simulated with a standard IEEE 802.11 DCF using the same packet size and transmission bitrate, in the same two-hop network with perfectly lossless links. This vividly demonstrates the significant performance penalty due to backoff and collisions caused by the random access scheme of IEEE 802.11, which are avoided with the explicit state synchronization mechanism, despite the overheads incurred.**

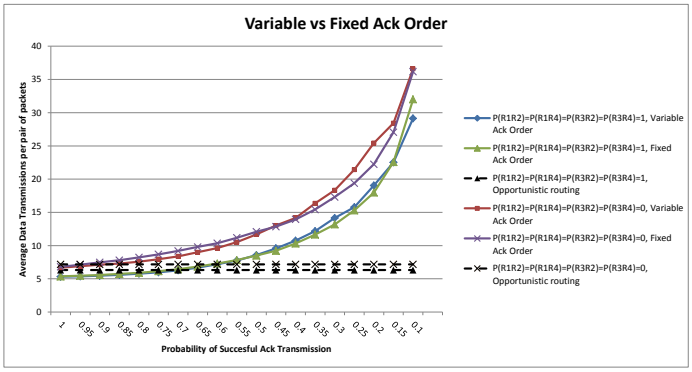
Next, we turn to evaluate the performance of the state synchronization mechanism in the multi-hop network case based on Figure 6, as discussed at the end of Section 5, for which the expected number of transmissions was 5.329 or 6.64, respectively, with or without overhearing across the dashed-line links. Before proceeding, we highlight the elevated importance played by the order of feed-

[6] More precisely, the MORE protocol, which is based on random linear coding of batches of packets, does require feedback from the receiver eventually once the entire batch can be decoded; however, the associated one-off overhead cost is amortized over all the packets in the batch and becomes negligible for large batch sizes.

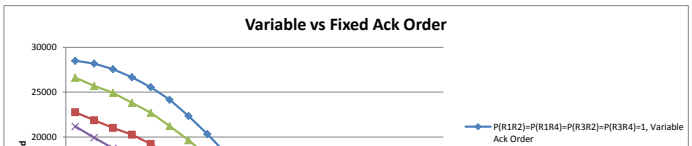
back transmissions in the ACK cycle in a multi-hop setting; indeed, when a feedback channel does not exist between some node pairs (i.e. nodes may not be able to overhear some of their peers' ACKs), a poorly chosen ACK transmission order will inevitably lead to an inconsistent view of the network state after every data packet transmission by certain nodes, causing frequent collisions or deadlocks that may lead to a performance collapse. Accordingly, in order to maximize the ability of all nodes to follow the network state, we propose to use a *variable* ACK order depending on the identity of the node that transmitted the data packet; specifically, we order the ACK transmissions by increasing distance from the data transmitter node (using the ETX metric distance for that purpose), so that even distant nodes may obtain an up-to-date view of the network state around the transmitter within a single ACK cycle. To ensure that every node is able to determine its position in the ACK cycle even if it did not overhear the data packet itself, we augment the ACK payload (Figure 7) to include the identity of the relevant node (whose data transmission triggered the ACK cycle). For ACK cycles started artificially when recovering from a deadlock or collision, we arbitrarily use the ACK order corresponding to a data transmission by node A.

The expected number of transmissions and the overall throughput as a function of the ACK reception probability are shown in Figure 9, **where, again, the dashed horizontal lines provide for comparison the corresponding values for opportunistic routing in both directions independently, which can be achieved using intra-flow coding**. Note that a comparison with the "self-state" feedback approach is omitted, since feedback messages reporting only the reception state of the node itself cannot be sensibly used for network state synchronization in a multi-hop situation where direct channels do not exist between many node pairs. Once again, we observe a graceful degradation of performance with increasing ACK loss rates, demonstrating the robustness of the state synchronization mechanism in the multi-hop case as well. For comparison, we also present the performance achieved when the transmission order in each ACK cycle is fixed, using the order A,B,R_1, R_2, R_3, R_4 regardless of the source of the data packet. With this order, certain transmission outcomes cannot be observed within the same ACK cycle even if the feedback channel is perfect in all existing links (e.g., when R_4 transmits a packet, R_1 and R_3 are unable to observe within a single ACK cycle whether the packet was successfully received by R_2). This leads to an inconsistent view of the network state after every such transmission, and causes a noticeable drop in throughput, as observed in Figure 9b, which reiterates the importance of a variable ACK order. This performance gap between a variable and a fixed ACK order gradually decreases with the ACK reception probability, but remains noticeable unless the ACK success probability drops to below 0.5, i.e. is worse than even the overhearing probability of the data packets, which, arguably, is unlikely to happen in a practical scenario.

To conclude this section, we emphasize that the description and simulations of the synchronization mechanism described herein have been provided for the purpose of giving insight into the impact of the synchronization overhead on the


Variable vs Fixed Ack Order
Average Data Transmissions per pair of packets
40
35
30
25
20
15
10
5
0
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
Probability of Succesful Ack Transmission
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=1, Variable Ack Order
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=1, Fixed Ack Order
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=1, Opportunistic routing
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=0, Variable Ack Order
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=0, Fixed Ack Order
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=0, Opportunistic routing

(a)


Variable vs Fixed Ack Order
30000
25000
20000
P(R1R2)=P(R1R4)=P(R3R2)=P(R3R4)=1, Variable Ack Order

performance of a possible MAC protocol implementing the optimal forwarding scheme; however, our description should not be considered to be a full-fledged MAC protocol proposal. There are many practical details that were left out from our discussion for simplicity, and would need to be addressed in a realistic protocol. For example, we assumed that the source nodes are always saturated with backlogged packets; hence, we did not consider how the relay nodes should distinguish in practice between a lost transmission and simply an empty queue at one of the sources (this could be achieved, for instance, by including an indication of the source's queue state in its own ACK message). Furthermore, we assumed an isolated system where all the nodes already know the link probability parameters in advance, and did not consider the questions of neighbor discovery and link estimation, or the effect of any possible interference from other nodes in the vicinity that may be competing for transmission at the same time. **Finally, there remains a broad scope for considering schemes that may allow the optimal bounds to be achieved with relaxed requirements for the network state feedback and topology information, e.g. incorporating a combination of *intra-flow* as well as inter-flow network coding [22, 23].** The design of a detailed MAC protocol based on the optimal forwarding concept and addressing the above practical issues remains a subject for future work.

7. Conclusion

We considered the problem of finding an optimal forwarding scheme that uses opportunistic routing and network coding to minimize the expected number of transmissions required to deliver a packet of data from every flow in the network. We presented a dynamic programming algorithm to calculate the optimal action for every possible network state (i.e. set of packets overheard by each node). Our performance evaluation demonstrated, for a simple example of a network with two nodes communicating via two common neighbors, that the optimal scheme can achieve a further reduction of up to 20% in the number of transmissions compared to either OR or NC employed in isolation, or a simple combination thereof, as proposed in existing literature. This superior performance is mainly due to two features that are absent from existing schemes: (1) the optimal scheme may base the decision of the next transmission on the state in all nodes, e.g. even if only one relay node has a coding opportunity, the optimal decision may depend on whether another node received any of the packets involved or none at all; and (2) the optimal scheme may involve opportunistic routing of the coded combination to some or all of the destinations, i.e. be opportunistic not only with respect to the choice of node to perform the network coding (if any), but the subsequent routing of the coded packet as well.

In this paper, we have not proposed any specific MAC protocol or mechanism that implements the optimal forwarding scheme. Rather, our goal has been to establish a theoretical framework to consider the fundamental performance limits of forwarding methods, and cast light on the classes of scenarios where existing schemes exhibit a sizeable performance gap from the optimum. We also illustrated the impact of the overhead from a possible robust feedback

mechanism to synchronize network states in all the nodes, and showed using simulation that the reduction in the number of transmissions of data packets brings about a significant improvement in throughput, even with the additional network state synchronization overhead. These encouraging results provide the motivation for further work towards the detailed design of fully distributed protocols based on the framework of this paper, which will need to address additional practical questions, such as scheduling in the presence of several flows competing for the same pool of relay nodes, and relay selection heuristics for dense or large-scale networks in which a full computation of the optimal forwarding scheme may be prohibitive.

Finally, we point out that the optimality definition employed in this study focused on delivering *one* packet in each direction of a unicast connection with 100% reliability using the least expected number of transmissions. In this formulation, once a packet from one side is delivered to its destination, the delivery of the opposite packet takes an absolute priority over any further packets from the same direction. A similar framework can be applied to other variations of the optimization target, e.g. to maximize the packet delivery ratio for a given 'budget' of transmissions. The adaptation of the dynamic programming solution to such alternative targets, as well as the design of a forwarding scheme that maximizes the network throughput region (i.e., the stability region of sustainable packet arrival rates to both endpoints) through the combination of OR and NC, remains an important topic of ongoing research [24].

[1] T. Mehmood and L. Libman. Towards optimal forwarding in wireless networks: Opportunistic routing meets network coding. In *Proc. IEEE LCN*, Zurich, Switzerland, October 2009.

[2] K.N. Ramachandaran, E.M. Belding, K.C. Almeroth, and M.M. Buddhikot. Interference-aware channel assignment in multi-radio wireless mesh networks. In *Proc. IEEE INFOCOM*, Barcelona, Spain, April 2006.

[3] K. Amouris. Space-time division multiple access (STDMA) and coordinated, power-aware MACA for mobile ad hoc networks. In *Proc. IEEE GLOBECOM*, pages 2890–2895, San Antonio, TX, November 2001.

[4] B. Aazhang, R.S. Blum, J.N. Laneman, K.J.R. Liu, W. Su, and A. Wittneben. Guest editorial: Cooperative communications and networking. *IEEE Journal on Selected Areas in Communications*, 25(2):241–244, February 2007.

[5] S. Biswas and R. Morris. ExOR: Opportunistic multi-hop routing for wireless networks. In *Proc. ACM SIGCOMM*, pages 133–144, Philadelphia, PA, August 2005.

[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. XORs in the air: Practical wireless network coding. *IEEE/ACM Trans. on Networking*, 16(3):497–510, June 2008.

[7] P. Larsson. Selection diversity forwarding in a multihop packet radio network with fading channel and capture. *ACM Mobile Computing and Communications Review*, 5(4):47–54, October 2001.

[8] S. Jain and S.R. Das. Exploiting path diversity in the link layer in wireless ad hoc networks. In *Proc. IEEE WoWMoM*, pages 22–30, Taormina, Italy, June 2005.

[9] M. Zorzi and R.R. Rao. Geographic random forwarding (GeRaF) for ad hoc and sensor networks: Energy and latency performance. *IEEE Trans. on Mobile Computing*, 2(4):349–365, October 2003.

[10] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proc. ACM SIGCOMM*, Kyoto, Japan, August 2007.

[11] Y. Lin, B. Li, and B. Liang. CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding. In *Proc. IEEE ICNP*, Orlando, FL, October 2008.

[12] Y.Y. Lin, C.C. Huang, and J.L. Huang. PipelineOR: A pipelined opportunistic routing protocol with network coding in wireless mesh networks. In *Proc. IEEE VTC*, Taipei, Taiwan, May 2010.

[13] A.F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros. Capacity of wireless erasure networks. *IEEE Trans. on Information Theory*, 52(3):789–804, March 2006.

[14] B. Ni, N. Santhapuri, Z. Zhong, and S. Nelakuditi. Routing with opportunistically coded exchanges in wireless mesh networks. In *Proc. IEEE WiMesh*, pages 157–159, Reston, VA, September 2006.

[15] J. Le, J.C.S. Lui, and D.M. Chiu. DCAR: Distributed coding-aware routing protocol for wireless networks. *IEEE Trans. on Mobile Computing*, 9(4):596–608, April 2010.

[16] J. Sun, Y. Liu, H. Hu, and D. Yuan. On-demand coding-aware routing in wireless mesh networks. *The Journal of China Universities of Posts and Telecommunications*, 17(5):80–86, October 2010.

[17] S. Sengupta, S. Rayanchu, and S. Banerjee. Network coding-aware routing in wireless networks. *IEEE/ACM Trans. on Networking*, 18(4):1158–1170, August 2010.

[18] J. Zhang, Y.P. Chen, and I. Marsic. Network coding via opportunistic forwarding in wireless mesh networks. In *Proc. IEEE WCNC*, pages 1775–1780, Las Vegas, NV, April 2008.

[19] Y. Yan, B. Zhang, H.T. Mouftah, and J. Ma. Practical coding-aware mechanism for opportunistic routing in wireless mesh networks. In *Proc. IEEE ICC*, pages 2871–2876, Beijing, China, May 2008.

[20] G.D. Forney Jr. The Viterbi algorithm. *Proc. of the IEEE*, 61(3):268–278, March 1973.

[21] K. Srinivasan, M. Jain, J. Choi, T. Azim, E.S. Kim, P. Levis, and B. Krishnamachari. The κ factor: Inferring protocol performance using inter-link reception correlation. In *Proc. ACM MobiCom*, pages 317–328, Chicago, IL, September 2010.

[22] H. Seferoglu, A. Markopoulou, and K.K. Ramakrishnan. I^2NC: Intra- and inter-session network coding for unicast flows in wireless networks. In *Proc. IEEE INFOCOM*, pages 1035–1043, Shanghai, P.R. China, April 2011.

[23] A. Khreishah, I.M. Khalil, and J. Wu. Polynomial time and provably efficient network coding scheme for lossy wireless networks. In *Proc. IEEE MASS*, pages 391–400, Valencia, Spain, October 2011.

[24] L. Libman, G.S. Paschos, L. Georgiadis, and X. Zhao. Throughput regions and optimal policies in wireless networks with opportunistic routing. In *Proc. International Symposium on Modeling and Optimization in Mobile, Ad hoc and Wireless Networks (WiOpt)*, Tsukuba, Japan, May 2013.