Universal Network Coding-Based Opportunistic Routing for Unicast

Abdallah Khreishah, *Member, IEEE*, Issa Khalil, *Member, IEEE* and Jie Wu, *Fellow, IEEE*

Abstract—Network coding-based opportunistic routing has emerged as an elegant way to optimize the capacity of lossy wireless multihop networks by reducing the amount of required feedback messages. Most of the works on network coding-based opportunistic routing in the literature assume that the links are independent. This assumption has been invalidated by the recent empirical studies that showed that the correlation among the links can be arbitrary. In this work, we show that the performance of network coding-based opportunistic routing is greatly impacted by the correlation among the links. We formulate the problem of maximizing the throughput while achieving fairness under arbitrary channel conditions, and we identify the structure of its optimal solution. As is typical in the literature, the optimal solution requires a large amount of immediate feedback messages, which is unrealistic. We propose the idea of performing network coding on the feedback messages and show that if the intermediate node waits until receiving only one feedback message from each next-hop node, the optimal level of network coding redundancy can be computed in a distributed manner. The coded feedback messages require a small amount of overhead, as they can be integrated with the packets. Our approach is also oblivious to losses and correlations among the links, as it optimizes the performance without the explicit knowledge of these two factors.

Index Terms—Network coding, wireless networks, cross-layer design, coded feedback, feedback, link correlation

1 Introduction

Designing efficient protocols for wireless multihop networks is not a straightforward extension from the protocols designed for their wireline counterparts, due to the unique properties of wireless links. One of the major properties of wireless links is the poor link quality. For example, recent studies [2] have shown that 50 percent of the operational links in Roofnet [3] have loss rates higher than 30 percent. Therefore, a major challenge for deploying wireless multihop networks is to design a transmission protocol that handles the lossy behavior of the wireless links efficiently.

An efficient way of handling losses in wireless multihop networks is to exploit the diversity among the links. Opportunistic routing [4] is the first trial to perform this exploitation. In opportunistic routing, there is no specific path from the source to the destination. Any node that overhears the packet can relay it. Take Fig. 1 as an example in which source node s wants to send packets to the destination d. The labels on the links represent their delivery rates. If we use traditional shortest path routing, the link between s and the chosen relay node will be the bottleneck, and the achievable rate will be upper bounded by 0.1. On the other hand, if we allow the node that receives the packet to forward it, the achievable rate will be $1-(1-0.1)^f$, which is a huge improvement over shortest path routing. The main challenge that faces the deployment of opportunistic routing is dealing with the case of when two relay nodes overhear the same packet. The work in [4] resolves this problem by assigning priorities to the next-hop forwarders, such that the node with higher priority will transmit first. All of the other next-hop forwarders have to listen to the transmission to decide whether one of the packets, overheard by a lower priority node, has been overheard by a higher priority node. If so, the lower priority node will not be responsible for forwarding the packet.

Performing opportunistic routing requires coordination among the links and the design of a specialized MAC protocol. It also requires all of the next-hop nodes to be able to overhear each other, which might not be available. In Fig. 2a source node s_1 wants to send packets to the destination node d_1 through the relay nodes v_1 and v_2. Assume that after the transmission of the a and b packets, node v_1 has a, and node v_2 has both a and b. Node v_2 has to know which packets to send, and how many of each type, based on overhearing the transmission of node v_1.

Intrasession network coding [5] can be used to resolve the shortcomings of opportunistic routing. In intrasession network coding, the source node divides the message it wants to send into batches, each having K packets of the form $P_1, \dots, P_K$. The source node keeps sending coded packets of the form $\sum_{i=1}^{K} \gamma_i P_i$, where $\gamma_i, \forall i$ is a random coefficient chosen over a finite field of a large enough size, typically 2^8–2^{16}. Upon receiving a coded packet, the intermediate relay node checks to see if the coded packet is linearly independent to what it has received before. If so, the node keeps the coded packet, otherwise it drops the packet. Each intermediate node generates linear combinations of the packets it has, and sends the resulting coded packets. When the destination receives K linearly independent packets, it can decode all of the packets of the batch. Therefore, it

- *A. Khreishah is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ07102. E-mail: abdallah@njit.edu.*
- *I. Khalil is with the Qatar Computer Research Institute, Doha, Qatar. E-mail: ikhalil@qf.org.qa.*
- *J. Wu is with the Department of Computer & Information Sciences, Temple University, Philadelphia, PA. E-mail: jiewu@temple.edu.*

Manuscript received 11 Mar. 2013; revised 2 May 2014; accepted 5 May 2014. Date of publication 7 May 2014; date of current version 8 May 2015.
Recommended for acceptance by L. Xiao.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2014.2322617

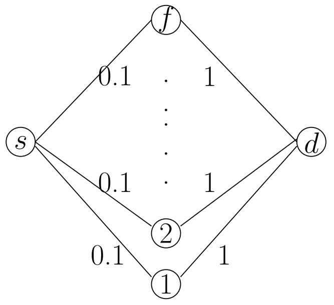
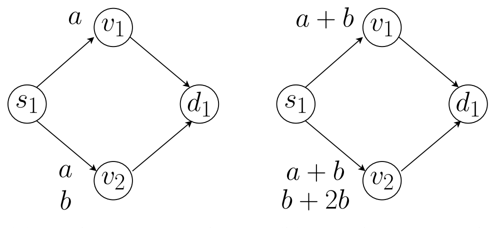


Fig. 1. An example of a network to illustrate opportunistic routing.

sends feedback to the source that requests it to stop sending from this batch and moves to the next batch. Intrasession network coding resolves the opportunistic routing problem due to the results in [6], which show that if the coding coefficients are chosen randomly over a large enough finite field, any two packets will be linearly independent with a very high probability. This property of random network coding eliminates the unnecessary feedback and overhearing requirements in opportunistic routing, and makes the design of the MAC layer independent of the other layers. In Fig. 2b the source node performs intrasession network coding. Instead of sending a it sends $a + b$, and instead of sending b it sends $a + 2b$. If the same reception scenario happens, node v_2 does not need to overhear node v_1 to make its own decision. It can generate a linear combination of the packets it has, say $2a + 3b$, which is linearly independent to the packet that node v_1 has. Therefore, if every relay node in this example delivers only one packet succesfully to the destination node, the whole batch (of size 2) can be recovered. This simple example shows that intrasession network coding transfers the problem from 'which and how many packets to send' to only 'how many packets to send', which is much easier to answer. The rest of the paper will be dedicated to answering this question optimally. Interested readers are encouraged to read the following references for more information about the use of network coding in wireless networks [7], [8], [9], [10].

Despite the simplicity that intrasession network coding creates for opportunistic routing, deciding the number of coded packets that each node has to send is a big challenge. The number of packets to be sent not only depends on the loss rates of the links, but also on the correlation among the links defined in [11]. To understand the challenge in choosing the number of transmitted packets, we provide an example that uses Fig. 3.



(a) Without network coding

(b) With network coding

Fig. 2. An example representing the effectiveness of network coding for opportunistic routing.

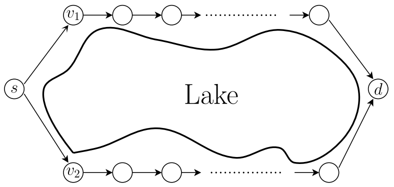
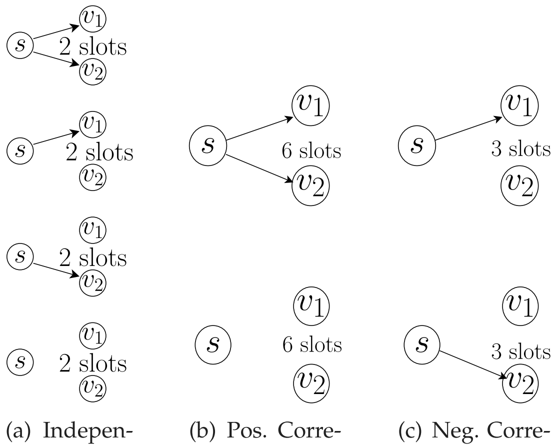


Fig. 3. A network with a lake in the middle, as used in the example.

In this example, node s is the source node and node d is the destination node. There are two paths that the packets can follow from the source to the destination, and these paths are separated by a lake. Therefore, nodes on one side of the lake cannot overhear nodes on the other side. Each of the two links, (s, v_1) and (s, v_2), has a delivery rate of 0.5, and we study three different cases as in [11]. *Case 1*: The two links are independent. This means that the reception process is independent among the links. *Case 2*: The two links are positively correlated. This means that if one link is inactive, the other one will be the same. *Case 3*: The two links are negatively correlated. This means that if one of the links is active, the other one will be inactive.

We use the following simple strategy. Each node stops the transmission of packets when it is sure that its next-hop nodes have collectively received the same number of linearly independent packets that it has received. For simplicity, we assume that the batch size is 6. For case 1, the source node needs to try for eight transmissions in order for both of the next-hop nodes to collectively receive the full rank, as illustrated in Fig. 4a. For case 2, the source node needs to try for 12 transmissions in order for both of the next-hop nodes to receive full rank. This is because both links will be active for the same six time slots and inactive in another six time slots, as illustrated in Fig. 4b. For case 3, the source node needs to try for six transmissions in order for both of the next-hop nodes to receive full rank. This is because the first link will be exclusively active for three time slots, and the second one will be exclusively active in the remaining three time slots, as illustrated in Fig. 4c. Therefore, if we follow the previously mentioned policy, the achievable throughput for case 3 will be



(a) Independent

(b) Pos. Correlated

(c) Neg. Correlated

Fig. 4. Illustration of the channel activation scenarios that insure that v_1 and v_2 collectively achieve full rank under different correlation conditions between the channels.

approximately 1.33 times that of case 1, and twice that of case 2. Note that the three rates can be made closer to each other if v_1 and v_2 stop sending packets when their next-hop nodes each receive three linearly independent packets. This means that they should be given a credit of only sending three linearly independent packets, regardless of the rank of the matrix that they have received.

Most of the previous works on opportunistic routing with intrasession network coding either assume that the links are independent and design the protocol based on that [5], [12], [13], [14], or use the forwarding rule that says the total number of received linearly independent packets should equal the number of linearly independent packets received by next-hop nodes [15]. In the above example, if we assume the first assumption, then, in case 2, there will not be enough linearly independent packets for the destination to decode because the source node did not send enough packets. Also, if we assume independent links in case 3, the throughput will be reduced by 1.33, due to the unnecessary transmissions. On the other hand, the works that design the rate control according to the rule that says the total number of received linearly independent packets should be the same as the ones received by next-hop nodes, result in throughput reduction in both cases 1 and 2.

In a general network, the links will have different correlations, and these correlations change over time, as is noted in [11]. This makes it difficult to perform measurements about the correlation, as to decide whether to use network coding or not. Therefore, it is crucial to design a strategy that guarantees a good performance in all cases and can adapt to the changes in the link qualities and the correlation among the links. To that end, we tackle the above problem in this paper and provide the following contributions:

- We formulate the problem of utility maximization for multiple unicast sessions that use network coding-based opportunistic routing on an arbitrary wireless multihop network, and use the duality approach to come up with the optimal distributed solution.
- We identify the challenges of implementing the optimal distributed algorithm to come up with a more practical algorithm. The practical algorithm works in a batch-by-batch manner and performs network coding on the feedback messages to exploit the broadcast nature of wireless links in the reverse direction. This reduces the number of feedback messages and eliminates the need for immediate feedback information. The algorithm is universal, as it takes into account the loss rates and the correlations among the links without the need to explicitly measure them.
- We prove that the batch-by-batch algorithm converges to the optimal solution.
- We present simulation results for our algorithm under different wireless settings, and show its superiority regardless of the channel's characteristics.

It should be noticed that there are other works that have considered the correlation among the links. The works in [16] and [17] solve a similar problem with a similar formulation. However, they assume a complete, immediate, and non-lossy feedback. Our paper resolves this very important practical problem by applying network coding to the feedback messages. We prove that by using this feedback mechanism, our solution converges to the optimal solution.

In the report [18], MORE can be extended to include the correlation among the links. However as shown in [16] Section 3.5, the objective of MORE is to minimize the number of transmissions not maximizing the throughput or achieving fairness. Also, in MORE, if the links are not independent, we need to measure the correlation among the links. This is different from our approach that converges to the optimal solution, without the need to explicitly know or measure the correlation among the links, which is not practical.

The work in [15] uses the coded feedback approach. However, the back-pressure algorithm they used is heuristic without any theoretical guarantee and does not take into account the correlations among the links. This differs from our approach that is proven to converge to the optimal solution.

2 Settings

In this paper, we consider a network represented by a set of nodes V. The links between the nodes are lossy and time-varying. A transmission by a node can be received by any subset of next-hop nodes. We represent this by a hyperedge (u, J), where u is the node that performs transmission, and J is a subset of the set of next-hop nodes. There are N unicast sessions in the network, each with a source s_i, a destination d_i, a rate R_i, and a utility function $U_i(R_i)$, $\forall i \in \{1, \dots, N\}$. Similar to most of the opportunistic routing protocols [5], [12], [14], [15], we are interested in the transmission of large files. Therefore, the throughput is the most important factor, and the individual packet delays are of no importance.

Since we are using intrasession network coding, one important factor to decide is the rate of linearly independent packets that a node has to successfully deliver to next-hop nodes. To model this factor, we use the concept of credits,[1] much like in [5], [16]. The symbol X^i_{uv} is used to represent the rate of credits transferred from node u to v for session i, which is the rate of linearly independent packets that node v has to deliver to next-hop nodes out of the linearly independent packets it has received from node u. Therefore, the total rate of credits for session i at node v would be $\sum_{u \in V} X^i_{uv}$, and these credits will be distributed to the next-hop nodes of v. We also use α^i_u to represent the fraction of time in which node u is scheduled to send the packets of session i. Symbol R_{uJ} represents the rate of packets that are sent by node u and are received by any of the nodes in the set J. Assume that there are K next-hop nodes for node u, then there are $2^K - 1$ combinations of next-hop nodes. Any hyperedge J here belongs to one of the $2^K - 1$ combinations. Symbol R_{uJ} here represents the rate of distinct packets sent by node u and received by the next-hop nodes in J. This depends on the

[1] There is a slight difference between the meaning of credit here and its meaning in [5]. In [5], it represents the number of linearly independent packets that have to be sent by a node. Here, it represents the number of linearly independent packets that have to be delivered successfully to next-hop nodes.

bandwidth of node u, the delivery rates of the links between u and each next-hop node in J and the correlation among these links.

3 Problem Formulation

Our problem can be formulated as follows. We want to maximize:

$$\sum_{i=1}^{N} U_i(R_i),$$

Subject to:

$$\sum_{v \in V} X_{vu}^i - \sum_{v \in V} X_{uv}^i \leq \begin{cases} -R_i, & u = s_i, \\ 0, & else, \end{cases} \qquad \forall i, \forall u \in V \backslash d_i \quad (1)$$

$$\sum_{v: v \in J} X_{uv}^i \leq \alpha_u^i R_{uJ} \qquad \forall i, \forall u \in V \backslash d_i, \forall (u, J). \quad (2)$$

We assume that the utility function $U_i(R_i)$ is nondecreasing and strictly concave. If the utility function is chosen properly, maximizing the objective function will achieve different kinds of fairness among the sessions [19]. Examples of $U_i(R_i)$ would be: $w_i \log(1 + R_i)$ and $w_i \frac{R_i^{1-\gamma}}{1-\gamma}$, where $0 \leq \gamma \leq 1$, and w_i is the weight assigned for session i.

Here, α_u^i depends on the underlying interference model. Typically, it corresponds to the convex hull of all of the achievable rates at all links [20]. Generally, the corresponding optimal scheduling policy is NP-hard [21] and approximation algorithms are used. In this paper, scheduling is of secondary importance, and we use the simple IEEE 802.11 protocol in the simulations. We use this so that we can focus on the network coding part, and so we can have a fair comparison with the other approaches that use the IEEE 802.11 protocol [5], [15].

Algorithm 1: Selecting the Packet to Send

```
$Z_v^i \leftarrow 0, \forall i, \forall v$
for $i \leftarrow 1$ to $N$ do
    Sort next-hop nodes according to $(q_u^i - q_v^i)$.
    Remove the nodes with negative backlog
    for Each sent packet $P$ do
        Choose node $v$ with the highest positive backlog
        difference such that $y_v^{iT} * P$ is zero.
        set $Z_v^i \leftarrow Z_v^i + 1$
    end for
    set $w_i \leftarrow \sum_v Z_v^i (q_u^i - q_v^i)^+$
end for
select $i^* \leftarrow \arg\ max_i w_i$.
send a packet from session $i^*$
```

α_u^i is a variable that depends on the MAC as well as the network layer's decisions. This is implied from the way the protocol works. Note that, working in a slot-by-slot manner, the value of $\alpha(t)$ will be either 0 or 1. If node u is scheduled and decides to send from session i using Algorithm 1, α_u^i at that time slot will be 1. Otherwise, it will be 0. If, we average $\alpha_u^i(t)$ over time, the result will be the optimal α_u^i.

The first set of constraints represents balance equations for the credits, so that the total received credits at a node should be equal to the total amount of sent credits. This guarantees that node d_i will receive linearly independent packets at a rate of no less than R_i. The second set of constraints represents the fact that if a packet is received by many nodes, only one of them can use this packet to increase its credits, which is a unique property of the wireless links.

Note that the constrains do not mean that the total number of sent linearly independent packets should be equal to the total received ones, due to the constraints set (2). For example, in the three cases in Fig. 3, if more than one node receives the packet, only one of them gets credit for that packet. Therefore, for each batch, the source node has six credits, and it distributes them evenly among its next-hop nodes. Even though both v_1 and v_2 receive linearly independent packets of rank 4, 6, 3, for cases 1, 2, 3, respectively, this also does not mean that the number of transmissions the source node has to make in the three cases should be the same. Therefore, the protocols that assume independent or negatively-correlated links perform poorly under other conditions.

According to the previous discussion, if there is only one session in the network, the achievable rate will be the max-flow between the source and the destination, regardless of the channel conditions. When more than one session exists in the network, then by the time-sharing variables α_u^i, the rate region, represented by the formulation, will be the maximum rate region that intrasession network coding can achieve. This is because intrasession network coding does not allow coding between different sessions; hence, the best thing to do is to allow time sharing among the sessions.

Since the constraints are linear, we have a convex optimization problem. The following proposition allows us to use the duality approach to solve the problem [22], [23].

Proposition 1. *Formulations (1)-(2) represent a convex optimization problem. Also, there is no duality gap between the primal and dual problems.*

Proof. The constraints represent an intersection of halfspaces, which represent a convex set. Also, we are maximizing a concave objective function. Therefore, the problem is a convex optimization problem. Due to the fact that the Slater conditions hold, there is no duality gap, i.e., the optimal solution to the dual problem is the same as the optimal solution to the primal problem. □

In the supplementary document, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2322617, we present a mapping from any wireless networks to an equivalent wireline network, such that the the capacity properties are preserved. With this conversion, we show that our formulation, achieves the maximum possible rate region with intrasession coding.

4 The Basic Algorithm

4.1 Structure of the Optimal Solution

Ignoring the scheduling constraints, we associate a Lagrange multiplier q_u^i with each constraint in (1). This results in the following Lagrange function:

$$L(\vec{R}, \vec{X}, \vec{q}) = \sum_{i=1}^{N} U_i(R_i) - q_{s_i}^i R_i - \sum_{i,u} q_u^i \Big(\sum_{v:v\in V} X_{vu}^i - \sum_{v:v\in V} X_{uv}^i \Big)$$

subject to (2).

After applying simple changes of variables, the Lagrange function becomes

$$L(\vec{R}, \vec{X}, \vec{q}) = \sum_{i=1}^{N} [U_i(R_i) - q_{s_i}^i R_i] + \sum_{u,i} \sum_{v} \big(q_u^i - q_v^i\big) X_{uv}^i$$

subject to (2).

The Lagrange function is separable [23], which means that the problem can be solved in a distributed way by using the gradient method, as follows.

Source algorithm. Each source s_i selects its rate at each time slot as follows:

$$R_i(t) = \arg\max_{R_i} [U_i(R_i) - q_{s_i}^i(t) R_i]. \tag{3}$$

Intermediate node algorithm. Each intermediate node u selects the number of credits for session i to transfer to all of its next-hop nodes, at each time slot, as follows:

$$\{X_{uv}^i(t)\}_{v\in V} = \arg\max_{\vec{X}} \sum_{v\in V} \Big(q_u^i(t) - q_v^i(t) X_{uv}^i \Big) \tag{4}$$

subject to:

$$\sum_{v\in J} X_{uv}^i \le \alpha_u^i R_{uJ} \qquad \forall i, \forall J. \tag{5}$$

Dual variables updates. The dual variables can be updated in a distributed way, as follows:

$$q_u^i(t+1) = \Bigg[q_u^i(t) + \beta_u^i \Big(\sum_{v\in V} X_{vu}^i(t) - \sum_{v\in V} X_{uv}^i(t) \Big) + R_i \times 1_{u=s_i} \Bigg]^+ \tag{6}$$

Here, $[.]^+$ is a projection on the positive real numbers, and β is the step size.

5 Our Practical Solution

In this section, we develop a practical protocol based on the structure of the optimal solution that we introduced in the previous section. To do so, we first specify the challenges of directly implementing the basic algorithm that was described in the previous section, and then we provide our practical solution.

5.1 Challenges

The Algorithm, represented by the operations in ((3)-(6)), converges to the optimal solution; however, it has the following shortcomings.

The first challenge is that the algorithm requires a large amount of feedback messages. For example, if a node that has l next-hop neighbors sends k packets from the batch, we need $(k \times l)$ feedback messages per batch. Also, the node that has l previous-hop neighbors needs to send different feedback messages to each one of these neighbors. Given that the wireless links are lossy further increases the challenges of the problem.

The second challenge is that the algorithm is based on slot-by-slot updates, which means that after sending a packet, a node has to get immediate and accurate feedback from all of the next-hop nodes, which is also impractical.

We resolve the first challenge by noting that the transmitted packets are coded packets. Therefore, we can compress the feedback into one coded packet that represents all of the received packets, which we will describe next. Therefore, we exploit the broadcast nature of wireless links in the reverse direction of transmission. We resolve the second challenge by performing the updates in a batch-by-batch manner instead of performing the updates on each time slot, as we will describe next.

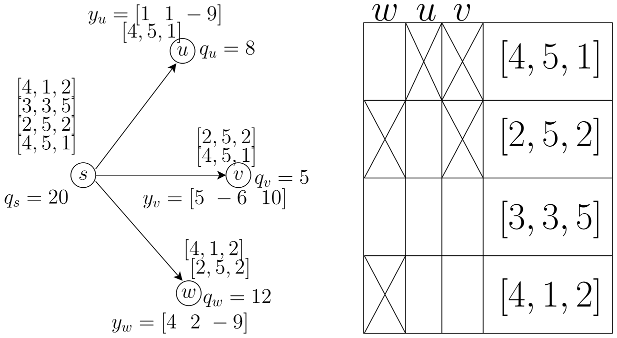
5.2 The Coded Feedback Approach

The coded feedback approach has been used previously by many works [15], [24], [25]. The work in [25] performed the coded feedback approach over multihop in wireline networks, i.e., network coding is performed on the feedback message, and these feedback messages are allowed to travel over more than one hop in the reverse paths between the source and the destination. The objective in [25] was to find the min-cut max-flow in wireline networks. For wireless networks, the works in [15], [24] use the coded feedback message at the immediate previous-hop node to perform rate control. In [24], the coded feedback approach is used for multicast, while in [15] it is used for the unicast case. In both cases, rate control was based on a heuristic, and no proof of the the optimality was provided. In this paper, we limit the coded feedback messages to be used by the previous one-hop-away nodes, much like [15], [24], but we show that our method achieves the optimal solution.

The common way of using the coded feedback is through the null space. The null space of the matrix A is the linear space of vectors such that the result of multiplying any one of them by A equals zero. For example, if y belongs to the null space of A, then $y^T A = 0$, where y^T is the transpose of y.

Take Fig. 5a as an example, in which node s sends four coded packets. Node v receives two of them. Node v can compute the null space of the space of the packets it receives, choose a vector from this space, and send it back to node s. As is illustrated in Fig. 5b: node s can now multiply this vector with each of the packets it has sent. If the result is zero, node s can infer that the packet has been received by node v with high probability. Otherwise, node s knows that the packet has not been received by node v. Using a hash table, the work in [15] makes the false-positive probability very low, about 10^{-10}.

Note that in the coded feedback approach, only one feedback message from the node can acknowledge all of its



(a) The transmission scenario

(b) The table created by the algorithm

Fig. 5. An example representing our coded feedback approach.

previous one-hop away nodes; therefore, it allows for the exploitation of the broadcast nature of wireless networks in the reverse direction of transmission. Also, as we will explain, these coded feedback messages can be integrated with the original packets with very low overhead.

5.3 Integrating the Coded Feedback Approach with the Algorithm

In this section, we move to apply the gradient algorithm that we adopted in the previous section in a batch-by-batch fashion. Therefore, the index t will refer to the batch number.

For the rate update (3) to be implemented in a batch-by-batch manner, the source will inject all of the packets of a given batch with the same rate specified by (3). Also, (4) and (6) can be done in a batch-by-batch manner by performing the transfer of the credits for a given batch, and the queue length updates at a node after making sure that the next-hop nodes have collectively received linearly independent packets equal to the number of credits for that batch at that node. After that, the node will move to the next batch. Therefore, what needs to be specified is how to use the coded feedback approach at the relay node u to perform the following two decisions that lead to maximizing (4), subject to (5).

- Node u has to decide the session that the current packet should be sent from.
- Node u has to also decide the number of credits to be assigned to each next-hop node.

To perform the first decision optimally, the relay node should choose session i^* that achieves the maximum value for the following among all of the sessions.

$$\left\{X^i_{uv}(t)\right\}_{v\in V} = \arg\max_{\vec{X}} \left(q^i_u(t) - q^i_v(t)\right) X^i_{uv} \qquad (7)$$

Subject to:

$$\sum_{v\in J} X^i_{uv} \leq \alpha^i_u R_{uJ} \qquad \forall (u,J). \qquad (8)$$

Note that in (7), the objective function is a linear function of X^i_{uv}. In order to perform maximization while satisfying (8), for every session i, node u ranks the next-hop nodes v according to the backlog difference $(q^i_u - q^i_v)$. Then it gives as many virtual credits[2] to this next-hop node with the highest backlog difference, subject to (8). Then it continues to do the same thing for the remaining nodes according to their backlog difference. For every sent packet, the next-hop node v gets a virtual credit if node v has received the packet, and no other node with a higher backlog difference has received the packet. This is because (8) means that, if more than one node receives a packet, only one of them can use that packet to increase its number of credits. Therefore, we give the credit to the node with the largest backlog. This process can be checked by using the coded feedback approach. Let us denote the number of virtual credits for session i and node v by Z^i_v; then node u calculates $w_i = \sum_v ((q^i_u - q^i_v) Z^i_v)$, such that all vs have positive backlog differences. Then node u selects the session that achieves the maximum w_i. Algorithm 1 describes the above strategy. In the algorithm, y^i_v represents a randomly selected vector from the null space of the ith session packet at node v. To a large extent, Algorithm 1 determines $\alpha^i_u(t)$.

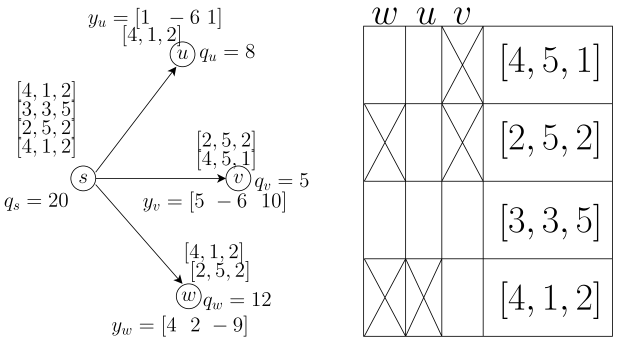
Every time a node receives a vector in the null space from the next-hop node, it multiplies that vector with all of the packets it has sent. This is done so that it can figure out the number of linearly independent packets that has been received by next-hop nodes. Once that rank becomes equal to the number of credits assigned for that batch at that node, the node distributes its credits to next-hop nodes in a fashion similar to that in Algorithm 1. However, this time the node only focuses on one session, and the credits that are assigned are real, not virtual credits. Algorithm 2 represents the credit assignment algorithm. In the algorithm, $TAKEN(P) = 1$ means that a credit for packet P has been assigned for one of the next-hop nodes that received packet P. Therefore, in line number 7 of the algorithm, no other next-hop node can be assigned a credit for this packet.

Algorithm 2: Credits Assignment

```
 1: Sort the next-hop nodes according to the backlog
    difference
 2: Discard the nodes with negative backlog difference.
 3: for Each sent packet $P$ do
 4:     $TAKEN(P) \leftarrow 0$
 5: end for
 6: for Each next-hop node $v$ with positive backlog
    difference in descending order do
 7:     for Each packet $P$ do
 8:         if $TAKEN(P) = 0$ AND $y^{iT}_v * P = 0$ then
 9:             $C^i_v \leftarrow C^i_v + 1$
10:             $TAKEN(P) \leftarrow 1$
11:         end if
12:     end for
13: end for
```

The example in Fig. 5a shows a broadcast link with one source node s and three receiving nodes, u, v, and w. The

[2] Note that these are different from the actual credits that will be distributed as a strategy for the second decision the node has to perform. These credits are just for knowing the packet of which session should be sent. Also, these credits are not transmitted to next-hop nodes. They are just computed locally.



w	u	v	
		☒	$[4, 5, 1]$
☒		☒	$[2, 5, 2]$
			$[3, 3, 5]$
☒	☒		$[4, 1, 2]$

(a) The transmission scenario

(b) The table created by the algorithm

Fig. 6. An example representing our coded feedback approach.

queue length of every node is represented in the figure. The figure also shows the four coded packets sent by s, and the coded packets received by the next-hop nodes. It also shows one vector in the null space[3] of the space of the received vectors by each node. Fig. 5b shows the result of multiplying the y vectors—which represent the null space of the received packets—with the packets at node s. If a cell in the table in Fig. 5b is filled, this means that the multiplication result of the packet in that row with the y vector of the next-hop node in that column is zero. This also means that the next-hop node in that column has overheard the packet in that row.

In Fig. 5a, the coded packets at the sender represent the sent packet from a specific batch and session, and the coded packets at the receivers are for the same session and batch. In Algorithm 1, node s first sorts the next-hop nodes according to the backlog difference in descending order which results in the following order, v, u, then w. Node s then multiplies the vectors in the null spaces of each one of these nodes, with each one of the sent packets. The result of the multiplication is illustrated in Fig. 5b. Then, node s assigns $Z_v^i = 2$ because node v has the highest backlog and has overheard two packets. Then node s assigns $Z_u^i = 0$, because all of the packets that are received by node u have been received by nodes with higher backlog differences. Using a similar approach, node w gets $Z_w^i = 1$. Therefore, the total weight obtained by Algorithm 1 for this session is $w_i = 38$. On the other hand, for the scenario in Fig. 6a, node s assigns: $Z_v^i = 2$, $Z_u^i = 1$, $Z_w^i = 0$, and $w_i = 42$.

In Fig. 5a, assume that node s has three credits to be distributed to next-hop nodes. Using the results on the table, the next-hop nodes of node s have collectively accumulated packets with ranks equal to the credits at the node. This can be verified from the table in Fig. 5b. Therefore, node s will assign the next-hop nodes the credits for this batch, and move on to the next one. Node v gets two credits, as it is the node with the highest backlog difference, and it has overheard two packets. Node u cannot be assigned a credit because the packet it has overheard has also been overheard by a node with a greater backlog difference. Therefore, the algorithm gives one credit to node w, due to the overhearing of the fourth packet, "$4X_1 + X_2 + 2X_3$." Fig. 6a represents the same example as Fig. 5a, but instead of making the links between s and u, v, respectively, positively correlated, we make them negatively correlated. Therefore, node u in this example receives "$4X_1 + X_2 + 2X_3$" instead of "$4X_1 + 5X_2 + X_3$." Since no credit has been assigned to node v for the packet "$4X_1 + X_2 + 2X_3$," node u can be assigned one credit. The examples represented by Figs. 5 and 6 show the adaptability of our algorithm to different channel conditions and correlations among links.

6 Evaluations

6.1 Simulation Settings

In this section, we evaluate the performance of our protocol in wireless networks with different network characteristics. We study the effect of varying different parameters on the performance of the network. These parameters are the loss rates of the links, the correlations among the links, and the number of sessions in the network. We are interested in the following two performance metrics: the total throughput observed by all of the sessions, and the fairness in allocating rates among the flows. We compare our protocol to two other protocols in the literature that represent the state-of-the-art opportunistic routing scheme with network coding. These are MORE [5] and CCACK [15]. In MORE, the authors assume that the links are independent. Based on that, the nodes in MORE periodically estimate the channels' loss rates, which allows the nodes to compute an estimation of the number of transmissions that each node has to perform. CCACK, on the other hand, uses the coded feedback approach. However, the rate control mechanism in CCACK is heuristic, and does not take into account the correlation among the links. We used MATLAB to perform the simulations in this section.

6.2 Results on an Illustrative Topology

We simulate one session using the topology in Fig. 3 with sixteen nodes. We vary two parameters: the delivery rates of all of the links, and the correlation between the links of the source node. The delivery rate values change from 0.3 to 0.8. For each one of these delivery rate values, we make the correlations between the links independent with $\kappa = 0$, as defined in [11], positively correlated with $\kappa = 1$, or negatively correlated with $\kappa = -1$. We assume very large files. Therefore, we run the simulations until the steady state throughput is reached, and then we record that value.

Fig. 7 illustrates our results. Our universal approach (UNIV) improves the throughput from 40 to 300 percent over both MORE and CCACK, depending on the scenario. The biggest improvement is noticed when the loss rates of the links are very high. This is due to the use of coded feedback in an optimal manner, which does not require too many feedback messages. The highest throughput is achieved by our protocol when the links are negatively correlated, whereas the lowest throughput is achieved when the links are positively correlated. This is due to the fact that the source node has to send more packets when the links are positively correlated, as explained in Section 1. Note that the difference between the positively correlated

[3] The coefficients for the y vectors here are over the real numbers just for illustration. When the algorithms are implemented, finite fields are used, and the negative value is changed depending on the size of the finite field.

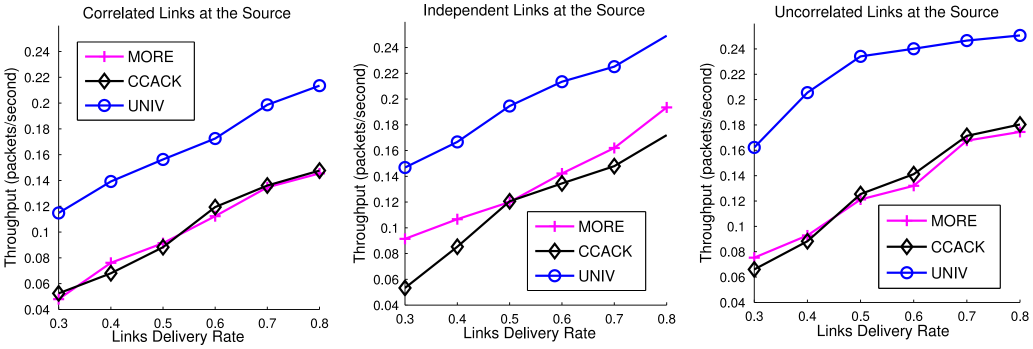


Fig. 7. Simulation results for the topology in Fig. 3 with one session.

and the negatively correlated cases is not large, because all of the nodes, except the source, are performing the same operations in both cases, as also explained in Section 1. The gain of our protocol is not only due to the use of the coded feedback approach, because CCACK, which uses this approach, performs similarly to MORE under these settings. In conclusion, the strength of our protocol lies behind integrating the coded feedback approach with cross-layer optimization.

Typically, the gain from network coding is higher when the links are negatively correlated. However, MORE performs slightly better when the links are independent. This is because MORE computes the number of packet transmissions that each node has to perform based on the assumption that they are independent. CCACK, on the other hand, achieves its highest performance when the links are negatively correlated, due to the use of the coded feedback approach.

We performed another set of simulations on the same topology in Fig. 3 with two opposite sessions, such that the source of one of them is the destination of the other one. The throughput results are similar to those with one session, as the two sessions share the network, and fair end-to-end rates were achieved by all of the schemes.

This motivates us to study bigger network topologies.

6.3 Results on Grid Topologies

We perform simulations on grid topologies of sizes 3×3 and 4×4. We change the delivery rates of the links in the range of 0.4 to 0.8, and the correlation coefficient among the links as defined in [11] in the range -1 to 1. We also change the number of unicast sessions from one to four sessions in the 3×3 topology, and from one to six sessions in the 4×4 topology. The results for the 3×3 topology are in Fig. 8, and the results for the 4×4 topology are in Fig. 9.

In the plots we measure either the average throughput or the average utility function value on different topologies that have the same parameter (the x-axis of the figure). Our utility function is $u_i = \log(1 + R_i)$, which is used to represent a combined factor of the total throughput and the utility as in [19], [26], [27].

As can be seen from both of the figures, the performance of our scheme increases with the number of sessions, while in the other two schemes, there is no significant performance increase when we increase the number of sessions. This is due to the session scheduling policy that we use. The utility or fairness index follows the same pattern, due to the same reason. The gain of our scheme is very low when we have one session in the network, and it becomes almost 100 percent when we increase the number of sessions. Another observation from the simulations is that our scheme takes advantage of the correlation among the links, as there is a difference between the case when $\gamma = -1$ and when $\gamma = -0.5$, while this is not noticeable for the other schemes. This is due to the credit assignment algorithm used. In the supplementary document, available online, we added more simulation results.

7 Conclusions and Future Work

In this paper, we have developed a distributed opportunistic routing algorithm that uses network coding. The design of our algorithm is inspired by recent results in the literature

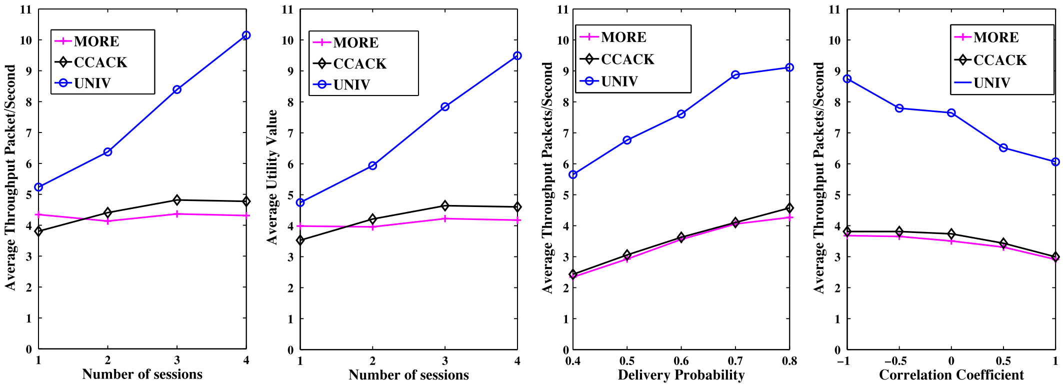


Fig. 8. Simulation results on a 3×3 grid.

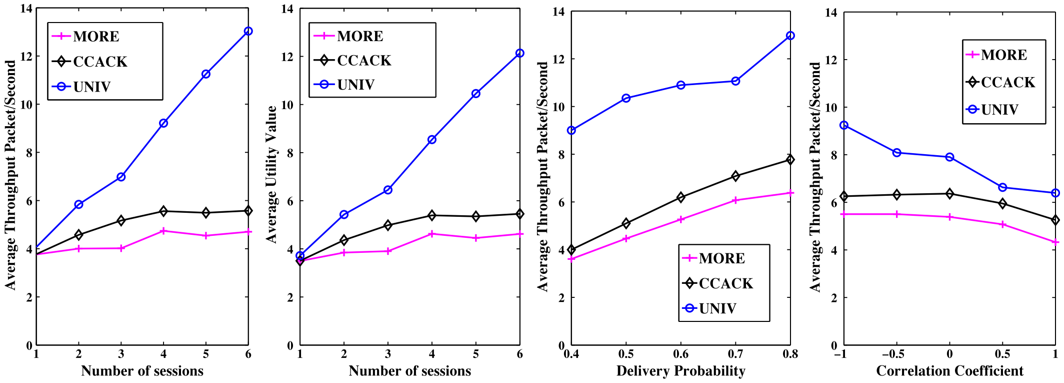


Fig. 9. Simulation results on a 4×4 grid.

that showed the sensitivity of the opportunistic routing and network coding protocols to the correlations among the wireless links. Our algorithm adapts to changes in the channel loss rates and the correlations among the links.

Acknowledgments

This research was supported in part by NSF grants ECCS 1331018, CNS 1065444, CCF 1028167, CNS 0948184, CCF 0830289, and Emirates Foundation grant EF 2009/075. Part of this work was presented in SECON 2012 [1].

References

[1] A. Khreishah, I. Khalil, and J. Wu, "Universal opportunistic routing scheme using network coding," in *Proc. IEEE Conf. Sensor, Mesh Ad Hoc Commun. Netw.*, Jun. 2012, pp. 353–361.

[2] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, Aug. 2004, pp. 121–132.

[3] MIT Roofnet. [Online]. Available: http://www.pdos.lcs.mit.edu/roofnet

[4] S. Biswas and R. Morris, "Opportunistic routing in multi-hop wireless networks," in *Proc. ACM Special Interest Group Data Commun.*, Philadelphia, PA, USA, Sep. 2005.

[5] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, Aug. 2007, pp. 169–180.

[6] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[7] C.-C. Wang, A. Khreishah, and N. Shroff, "On cross-layer optimizations for intersession network coding on practical 2-hop relay networks," in *Proc. Conf. Rec. 43rd Asilomar Conf. Signals, Syst. Comput.*, Nov. 2009, pp. 771–775.

[8] A. Khreishah, I. Khalil, and J. Wu, "Distributed network coding-based opportunistic routing for multicast," in *Proc. 13th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2012, pp. 115–124.

[9] A. Khreishah, I. Khalil, P. Ostovari, and J. Wu, "Flow-based XOR network coding for lossy wireless networks," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 2321–2329, Jun. 2012.

[10] P. Ostovari, J. Wu, and A. Khreishah, "Network coding techniques for wireless and sensor networks," *The Art Wireless Sensor Netw.*, vol. 1, pp. 129–162, 2013.

[11] K. Srinivasan, M. Jain, J. Choi, T. Azim, E. Kim, P. Levis, and B. Krishnamachari, "The κ-factor: Inferring protocol performance using inter-link reception correlation," presented at the ACM 19th Annu. Int. Conf. Mobile Comput. Netw., Chicago, IL, USA, Sep. 2010.

[12] B. L. Y. Lin and B. Liang, "CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding," in *Proc. 16th IEEE Int. Conf. Netw. Protocols*, Oct. 2008, pp. 13–22.

[13] C. Gkantsidis, B. Radunovic, P. Key, S. Gheorgiu, W. Hu, and P. Rodriguez, "Multipath code casting for wireless mesh networks," in *Proc. ACM Int. Conf. Emerging Netw. Experiments Technol. Conf.*, 2007, p. 10.

[14] X. Zhang, and B. Li, "Optimized multipath network coding in lossy wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 622–634, Jun. 2009.

[15] D. Koutsonikolas, C.-C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *Proc. 29th Conf. Comput. Commun.*, San Diego, CA, USA, Mar. 2010, pp. 1–9.

[16] B. Radunovic, C. Gkantsidis, P. Key, and P. Rodriguez, "Toward practical opportunistic routing with intra-session network coding for mesh networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 420–433, Apr. 2010.

[17] D. Lun, M. Medard, and R. Koetter, "Network coding for efficient wireless unicast," in *Proc. Int. Zurich Seminar Commun.*, 2006, pp. 74–77.

[18] S. Chachulski, "Trading structure for randomness in wireless opportunistic," M.S. thesis, Massachusetts Inst. Technol., Dept. Electrical Eng. Comput. Sci., Cambridge, MA, USA, May 2007.

[19] T. Bonald and L. Massoulie, "Impact of fairness on Internet performance," in *Proc. ACM Joint Int. Conf. Meas. Model. Comput. Syst.*, Cambridge, MA, USA, Jun. 2001, pp. 82–91.

[20] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.

[21] T. Cui, L. Chen, and T. Ho, "Distributed optimization in wireless networks using broadcast advantage," in *Proc. IEEE Conf. Decision Control*, Dec. 2007, pp. 5839–5844.

[22] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[23] D. Bertsekas and J. N. Tsitsikalis, *Parallel and Distributed Computation: Numerical Methods*. Belmont, MA, USA: Athena Scientific, 1997.

[24] J. Park, M. Gerla, D. Lun, Y. Yunjung, and M. Medard, "Codecast: A network-coding-based ad hoc multicast protocol," *IEEE Wireless Commun.*, vol. 13, no. 5, pp. 76–81, Oct. 2006.

[25] C.-C. Wang, "Pruning network coding traffic by network coding—A new class of max-flow algorithms," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1909–1929, Apr. 2010.

[26] A. Khreishah, C.-C. Wang, and N. Shroff, "Rate control with pairwise intersession network coding," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 816–829, Jun. 2010.

[27] X. Lin and N. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. 43rd IEEE Conf. Decision Control*, Dec. 2004, pp. 1484–1489.

Abdallah Khreishah received the BS degree with honors from Jordan University of Science & Technology in 2004. He received the PhD and MS degrees in electrical and computer engineering from Purdue University in 2010 and 2006, respectively. In Fall 2012, he joined the ECE department of New Jersey Institute of Technology as an assistant professor. His research spans the areas of network coding, wireless networks, congestion control, cloud computing, and network security. He is a member of the IEEE.

Issa Khalil received the BSc and MS degrees from Jordan University of Science and Technology in 1994 and 1996, and the PhD degree from Purdue University in 2006, all in computer engineering. He joined the Faculty of Information Technology (FIT) of the United Arab Emirates University (UAEU) in August 2007, where he was promoted to an associate professor in September 2011. In August 2012, he was appointed as the chair of the Information Security department at UAEU. In June 2013, he joined the Qatar Computing Research Institute (QCRI) as a senior scientist in the cyber security group. His research interests span the areas of wireless and wireline communication networks. He is especially interested in security, routing, and performance of wireless Sensor, Ad Hoc, and Mesh networks. He is a member of the IEEE.

Jie Wu is the chair and Laura H. Carnell professor in the Department of Computer and Information Sciences, Temple University. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His research interests include wireless networks and mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He has published more than 600 papers in various journals, conference proceedings, and books. He serves on the editorial board of the *IEEE Transactions on Computers* and the *Journal of Parallel and Distributed Computing*. He has served as an IEEE computer society distinguished visitor. Currently, he is the chair of the IEEE Technical Committee on Distributed Processing (TCDP) and ACM distinguished speaker. He is a fellow of the IEEE.