2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)

Universal Opportunistic Routing Scheme using Network Coding

Abdallah Khreishah[†], Issa M. Khalil[‡], and Jie Wu[†]
[†]Department of Computer & Information Sciences Temple University, Philadelphia, PA 19122
[‡]Faculty of Information Technology, United Arab Emirates University, UAE

***Abstract*—Recent research has shown that the performance of opportunistic routing and network coding in wireless networks is greatly impacted by the correlation among the links. However, it is difficult to measure the correlation among the links, especially because of the time-varying behavior of the wireless links. Therefore, it is crucial to design a distributed algorithm that does not require the explicit knowledge of the channels' states and can adapt to the varying channel conditions. In this paper, 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.**

I. Introduction

Wireless multihop networks play major roles in almost every field of our lives. One of the unique properties of wireless links is the poor link quality. For example, recent studies [1] have shown that 50% of the operational links in Roofnet [2] have loss rates higher than 30%. 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 [3] 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 [3] 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, as shown in Fig. 2. Intrasession network coding [4] 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 $P_1,\ldots,P_K$. 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 sends feedback to the source that requests it to stop sending from this batch and move to the next batch. Intrasession network coding resolves the opportunistic routing problem due to the results in [5] 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.

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 [6]. To understand the challenge in choosing the number of transmitted packets, we provide an example that uses Fig. 2.

978-1-4673-1905-8/12/$31.00 ©2012 IEEE

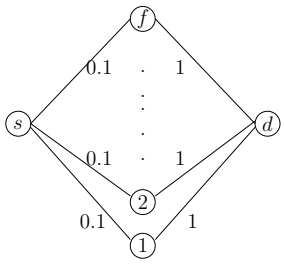


Fig. 1. An example of a network to illustrate opportunistic routing.

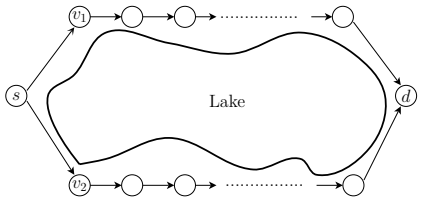


Fig. 2. A network with a lake in the middle used in the example.

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 [6]. **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 or correlated as termed in [6]. This means that if one link is inactive, the other one will be the same. **Case 3**: The two links are negatively correlated or uncorrelated as termed in [6]. This means that if one of the links is active, the other one will be inactive.

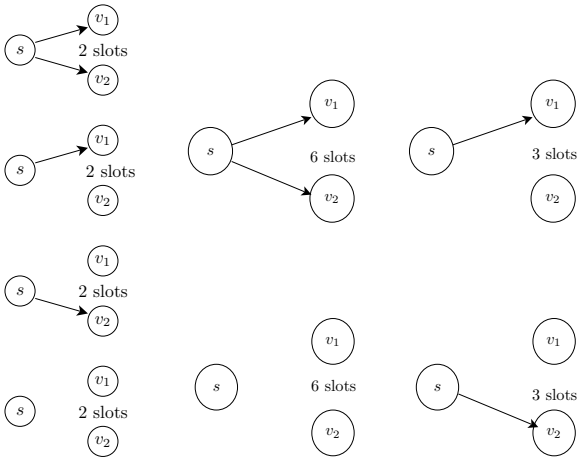
We use the following simple strategy. Each node stops the transmission of packets when it is sure that its next-hop nodes have collectively received the same number of linearly independent packets to what it has received. For simplicity, we assume that the batch size is 6. For case 1, the source node needs to try for 8 transmissions in order for both of the next-hop nodes to collectively receive the full rank, as illustrated in Fig. 3(a). 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 6 time slots and inactive in another 6 time slots, as illustrated in Fig. 3(b). For case 3, the source node needs to try for 6 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 3 time slots, and the second one will be exclusively active in the remaining 3 time slots, as illustrated in Fig. 3(c). Therefore, if we follow the previously mentioned policy, the achievable throughput for case 3 will be approximately 1.33 times 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 3 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 work on opportunistic routing with intrasession network coding either assume that the links are independent and design the protocol based on that [4], [7]–[9], 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 [10]. 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 number of packets sent by the source node is not enough. 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 [6]. This makes it difficult to perform measurements about the correlation 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 uses 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 onto 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.

The rest of the paper is organized as follows. In Section II, we describe our settings. We then formulate the problem in Section III. The structure of the optimal solution is described in Section IV. In section V, we outline the coded feedback-



(a) Independent (b) Correlated (c) Uncorrelated

Fig. 3. 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.

based backpressure algorithm, followed by the performance analysis in Section VI. We present the experiment results in Section VII and conclude the paper in Section VIII.

II. 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 $U_i(R_i)$, $\forall i \in \{1, \ldots, N\}$. Similar to the most of the opportunistic routing protocols [4], [7], [9], [10], 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 to [4], [11]. 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 independents 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.

[1]There is a slight difference between the meaning of credit here and its meaning in [4]. In [4], 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.

III. Problem Formulation

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

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

Subject to:

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

$$\sum_{v : v \in J} X^i_{uv} \leq \alpha^i_u R_{uJ} \qquad \forall i, \forall u \in V \backslash d_i, \forall (u, J) \tag{2}$$

We assume that the utility function $U_i(R_i)$ is non-decreasing and strictly concave. If the utility function is chosen properly, maximizing the objective function will achieve different kinds of fairness among the sessions [12]. 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^i_u depends on the underlying interference model. Typically, it corresponds to the convex hull of all of the achievable rates at all links [13]. Generally, the corresponding optimal scheduling policy is NP-hard [14], [15] 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 to have a fair comparison with the other approaches that use the IEEE 802.11 protocol [4], [10].

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 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. 2, 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 6 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, respectively, 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 uncorrelated 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 between 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 [16], [17].

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. ■

IV. The Basic Algorithm

A. 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

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

subject to (2).

The Lagrange function is separable [17], 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:

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

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

V. 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.

A. Challenges

The Algorithm, represented by the operations in ((3)-(6)), converges to the optimal solution, but 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.

B. The Coded Feedback Approach

The coded feedback approach has been used previously by many works [10], [18], [19]. The work in [19] 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 reverese paths between the source and the destination. The objective in [19] was to find the min-cut max-flow in wireline networks. For wireless networks, the works in [10], [18] use the coded feedback message at the immediate

previous-hop node to perform rate control. In [18], the coded feedback approach is used for multicast, while in [10] 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 [10], [18], 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 anyone 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. 4(a) 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. 4(b): 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 been not received by node v. Using a hash table, the work in [10] 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 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.

C. 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 bach 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 decide also 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.

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

Subject to:

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

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, 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 i-th session packets at node v.

Algorithm 1 Selecting the packet to send

```
Z^i_v ← 0, ∀i, ∀v
for i ← 1 to N do
    Sort next-hop nodes according to (q^i_u − q^i_v).
    Remove the nodes with negative backlog
    for Each sent packet P do
        Choose node v with the highest non-negative backlog
        difference such that y^{iT}_v * P is zero.
        set Z^i_v ← Z^i_v + 1
    set w_i ← Σ_v Z^i_v (q^i_u − q^i_v)^+
select i* ← arg max_i w_i.
send a packet from session i*
```

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 to figure out the number of linearly independet 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 Algorithm 1. However, this time the node only focuses on one session,

[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.

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: **for** Each next-hop node v with positive backlog difference in descending order **do**
6: **for** Each packet P **do**
7: **if** $TAKEN(P) = 0$ AND $y_v^{iT} * P = 0$ **then**
8: $C_v^i \leftarrow C_v^i + 1$
9: $TAKEN(P) \leftarrow 1$

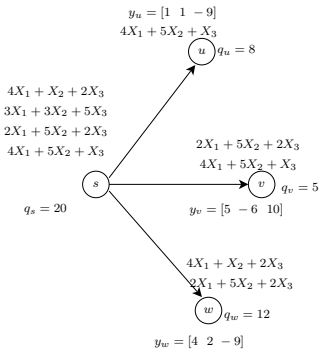
and the credits that are assigned are real, not virtual credits. Algorithm 2 represents the credit assignment algorithm. In the algorithm, $TAKEN(P) = 1$ means that a credit for packet P has been assigned for one of the next-hop nodes that received packet P. Therefore, in line number 7 of the algorithm, no other next-hop node can be assigned a credit for this packet.

The example in Fig. 4(a) shows a broadcast link with one source node s and three receiving nodes, u, v, and w. The queue length of every node is represented in the figure. The figure also shows the 4 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. 4(b) 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. 4(b) 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.

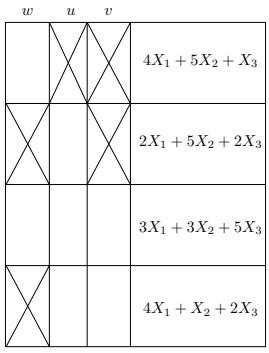
In Fig. 4(a), the coded packets at the sender represent the sent packet from a specific batch and session, and the coded packets at the receivers are for the same session and batch. In Algorithm 1, node s first sorts the next-hop nodes according to the backlog difference in descending order, which results 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. 4(b). 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. 5(a), node s assigns: $Z_v^i = 2$, $Z_u^i = 1$, $Z_w^i = 0$, and $w_i = 42$.

In Fig. 4(a), assume that node s has three credits to be distributed to next-hop nodes. Using the results on the table,

[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 minus value is changed depending on the size of the finite field.

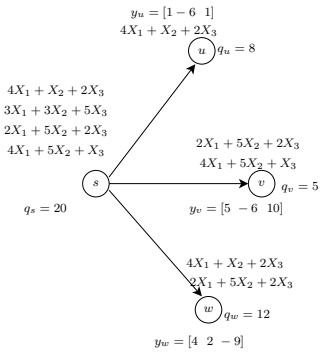


(a) The transmission scenario

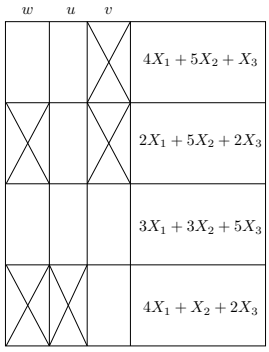


(b) The table created by the algorithm

Fig. 4. An example representing our coded feedback approach.



(a) The transmission scenario



(b) The table created by the algorithm

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

the next-hop nodes of node s have collectively accumulated packets with rank equal to the credits at the node. This can be verified from the table in Fig. 4(b). Therefore, node s will assign the next-hop nodes the credits for this batch and move on to the next one. Node v gets 2 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 been also 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. 5(a) represents the same example as Fig. 4(a), but instead of making the links between s and u, v, respectively, correlated, we make them uncorrelated. 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. 4 and 5 show the adaptability of our algorithm to different channel conditions and correlations among links.

D. Details of the Practical Protocol

So far, we have identified the structure of the optimal solution and discussed the major challenges that face its deployment. We have then designed a back-pressure algorithm that uses the coded feedback approach and works in a batch-

by-batch manner to resolve these challenges. In this section, we outline the details of the practical protocol that uses the insight drawn from the structure of the optimal solution and resolves the above-mentioned challenges.

In our protocol, every node maintains the following information. The received and sent coded packets, the available number of credits, and the batch and session numbers of the received and sent packets. We adopt a packet format that is similar to [4], [10] in that each packet has a 1500 bytes of data. The packets also contain the coefficients of the coding vector along with its session and batch numbers. The packet contains the three most recent batches it has received each with a vector from the null space of the packets in that batch. The packet contains the number of currently queued credits. The packets also contain the number of credits assigned to each next-hop node and the batch number for these credits. Fig. 6 represents the packet format. In conclusion, the packet format allows both (1) the feedback messages to previous-hop nodes, and (2) the credits assignment messages to next-hop nodes to be integrated with the packets with less than 3% overhead.

VI. Convergence Analysis

In this section, we show that our batch-based algorithm converges to the optimal solution. We do that by first presenting the structure of the dual problem, and then showing that the dual variables converge to their optimal value; finally, we show that objective function of the problem converges to its optimal value.

Note that the Lagrange function can be rewritten as:

$$L(\vec{R},\vec{X},\vec{q}) = \sum_{i=1}^{N}[U_i(R_i) - q^i_{s_i}R_i] + \sum_{\substack{u,i\\ \vec{X}\in Y}}\sum_{v}(q^i_u - q^i_v)X^i_{uv},$$

where Y is the feasible set that is represented by (2).

We can define $D(\vec{q}) = \max_{\vec{R},\vec{X}\in Y} L(\vec{R},\vec{X},\vec{q})$; hence, the dual problem becomes: $\min_{\vec{q}\geq 0} D(\vec{q})$.

Note that as we increase the batch size, Algorithm 2 computes (4), subject to (5), more accurately. The reason is that there is a delay between the time the next-hop nodes collectively receive the required number of linearly independent packets and the time when node u is notified of this event. However, as we increase the batch size, the effect of this delay decreases until reaching almost zero.

We have the following Proposition:

Proposition 2: Let the step size β^i_u be a fixed constant number β for all batches, and let ϕ be the set of positive $\vec{q}$ that minimizes $D(\vec{q})$, and $d(\vec{q},\phi) = \min_{\vec{q^*}\in\phi} ||\vec{q}-\vec{q^*}||$. Given any $\epsilon > 0$, $\exists\ \beta_0 > 0$ such that $\forall \beta < \beta_0$ and any initial $\vec{q}(0)$, $\exists\ t_0$ such that $\forall t > t_0$ we have:

$$d(\vec{q}(t),\phi) < \epsilon$$

Proof: Note that the dual problem is a convex one. Also, the absolute value of the gradient of the dual problem is bounded by $\sum_{u,i}(\sum_v X^i_{vu} - \sum_v X^i_{uv})$, which is bounded by $\sum_{u,i}\alpha^i_u BW_u < \infty$, where BW_u is the bandwidth of node u. Therefore, by Lemma 8.2.1 and Proposition 8.2.2 from [20], our results follow. ■

The previous proposition shows that the dual variables will converge to their optimal values. It also shows that the queue lengths will be finite and the system will be stable. The next theorem establishes the optimality of our algorithm with respect to the objective function.

Theorem 1: Let the step size β^i_u be a fixed constant number β for all batches, and let U^* be the optimal solution to our problem ((1)-(2)). Given any $\epsilon > 0$, $\exists\beta_0$ s.t. $\forall\ \beta < \beta_0$, we have:

$$\lim\inf_{m\to\infty}\sum_{t=1}^{m}\sum_i U_i(R_i(t) > U^* - \epsilon$$

Proof: We use the following Lyapunov function:

$$V(\vec{q}(t)) = \frac{1}{2}\sum_{u,i}(q^i_u(t))^2$$

We have:

$$V(\vec{q}(t+1)) - V(\vec{q}(t)) \leq$$
$$\sum_i q^i_{s_i}(t)R_i(t) + \beta\sum_{\substack{u,i\\ \vec{X}\in Y}} q^i_u(t)(\sum_v X^i_{vu}(t) - \sum_v X^i_{uv}(t))$$
$$+ \beta^2 A(t)$$

where:

$$A(t) = \sum_{\substack{u,i\\ \vec{X}\in Y}}(\sum_v X^i_{vu}(t) - \sum_v X^i_{uv}(t))^2$$
$$\leq \sum_{u,i}(\alpha^i_u BW_u)^2 = \bar{A}$$

After multiplying both sides of the inequality by -1 and adding $\beta\sum_i U_i(R_i(t))$, we have:

$$V(\vec{q}(t)) - V(\vec{q}(t+1)) + \beta\sum_i U_i(R_i(t))$$
$$\geq \beta[\sum_i U_i(R_i(t)) - q^i_{s_i}(t)R_i(t)$$
$$- \sum_{\substack{u,i\\ \vec{X}\in Y}} q^i_u(t)(\sum_v X^i_{vu}(t) - \sum_v X^i_{uv}(t))] - \beta^2\bar{A}$$
$$= \beta D(\vec{q}(t)) - \beta^2\bar{A}$$
$$\geq \beta U^* - \beta^2\bar{A}$$

This is due to the duality theorem [16].

After summing up both sides of the above inequality over $t = 1,\ldots,M$ and dividing both sides by M, we have:

$$\frac{\beta}{M}\sum_{t=1}^{M}\sum_i U_i(R_i(t))$$
$$\geq \beta U^* - \beta^2\bar{A} - \frac{V(\vec{q}(1)) - V(\vec{q}(M+1))}{M}$$

Data	Coding Coefficients	Session Number	# of queued credits	Most recent batch	Null space vector	.....	M-th most recent batch	Null space vector	Batch ID of next-hop credits	# of credits for first next-hop	.....	# of credits for last next-hop

Fig. 6. Representation of the packets format in our protocol.

Note that all $V(.)$ are bounded because the sequence $\{\vec{q}(t)\}$ converges. For any given $\epsilon > 0$, the term $\frac{V(\vec{q}(1))-V(\vec{q}(M+1))}{\beta M}$ can be bounded by $\frac{\epsilon}{2}$ for a large enough M. Also, for any given $\epsilon > 0$, $\exists, \beta_1 > 0$, such that $\beta \bar{A} \leq \frac{\epsilon}{2}$, $\forall \beta \leq \beta_1$. Therefore, we have:

$$\lim \inf_{m \to \infty} \sum_{t=1}^{m} \sum_{i} U_i(R_i(t) > U^* - \epsilon$$

■

VII. Evaluations

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 [4] and CCACK [10]. 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.

We simulate one session using the topology in Fig. 2 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 [6], correlated with $\kappa = 1$, or uncorrelated 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 by 40% to 300% 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 uncorrelated, whereas the lowest throughput is achieved when the links are correlated. This is due to the fact that the source node has to send more packets when the links are correlated, as explained in Section I. Note that the difference between the correlated and uncorrelated 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 I. The gain of our protocol is not only due to the use of the coded feedback approach, because CCACK that uses this approach performs similar 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 uncorrelated. 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 uncorrelated due to the use of the coded feedback approach.

We performed another set of simulations on the same topology in Fig. 2 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.

VIII. 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 the recent results in the literature that showed the sensitivity of the opportunistic routing and network coding protocols to the correlations among the wireless links. We designed our algorithm based on formulating the problem with arbitrary channel conditions as a convex optimization problem, which results in an optimal back-pressure algorithm. We identified the challenges of implementing the optimal back-pressure algorithm. This leads us to use the coded feedback approach to resolve the deployment difficulties. Our algorithm adapts to changes in the channel loss rates and the correlations among the links. We showed the advantage of our approach through simulation results.

Our future research will be directed toward:

- Extending our results for intersession network coding, where coding can be performed among the packets of different multicast sessions.
- Modifying the Transmission Control Protocol (TCP) to work with our protocol with the possibility of sending

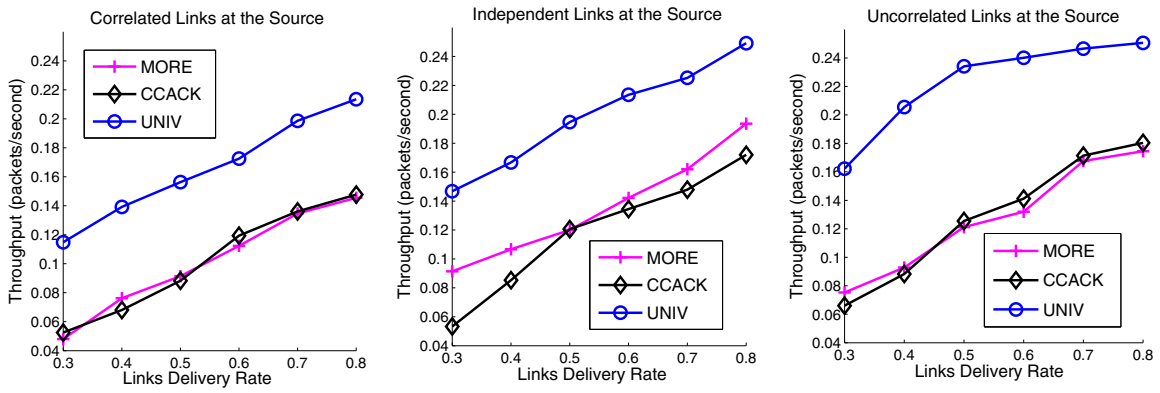


Fig. 7. Simulation results for the topology in Fig. 2 with one session.

coded end-to-end coded feedback messages to control the window size and other parameters in TCP.

IX. Acknowledgment

This research was supported in part by NSF grants ECCS 1128209, CNS 1065444, CCF 1028167, CNS 0948184, and CCF 0830289.

The authors would like to thank Dr. Shan Lin for his insightful comments and Pouya Ostovari for the help with running the simulations.

References

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network,," in *ACM SIGCOMM,*, Aug 2004.

[2] "MIT Roofnet, http://www.pdos.lcs.mit.edu/roofnet."

[3] S. Biswas and R. Morris, "Opportunistic routing in multi-hop wireless networks," in *Proc. ACM Special Interest Group on Data Commun. (SIGCOMM), Philadelphia, PA, USA*, Sept 2005.

[4] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *ACM Special Interest Group on Data Commun. (SIGCOMM) Kyoto, Japan,*, Aug 2007.

[5] 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. Inform. Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[6] 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,," in *ACM MobiCom. Chicago, IL,*, Sept 2010.

[7] B. L. Y. Lin and B. Liang, "CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding," in *in the Proceedings of the 16th IEEE International Conference on Network Protocols (ICNP 2008), Orlando, Florida*, October 2008.

[8] C. G. et al., "Multipath code casting for wireless mesh networks," in *in Proc. of ACM CoNEXT*, 2007.

[9] X. Zhang and B. Li, "Optimized multipath network coding in lossy wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 5, pp. 622–634, 2009.

[10] D. Koutsonikolas, C.-C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *in Proceedings of the 29th Conference on Computer Communications (INFOCOM), San Diego, USA,*, March 2010.

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

[12] T. Bonald and L. Massoulie, "Impact of fairness on Internet performance," in *Proc. of ACM Joint Int'l Conf. on Measurement and Modeling of Computer Systems (Sigmetrics), Cambridge, MA*, June 2001.

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

[14] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "On the complexity of scheduling in wireless networks," in *proceedings of ACM MOBICOM*, Sept 2006.

[15] T. Cui, L. Chen, and T. Ho, "Distributed optimization in wireless networks using broadcast advantage," in *IEEE Conf. Decision and Contr. New Orleans,*, Dec 2007.

[16] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

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

[18] J. Park, M. Gerla, D. Lun, Y. Yunjung, and M. Medard, "Codecast: a network-coding-based ad hoc multicast protocol," *IEEE Wireless Communications*, vol. 13, no. 5, 2006.

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

[20] D. Bertsekas, A. Nedić, and A. Ozdaglar, *Convex Analysis and optimization*. Athena Scientific, 2003.