Optimized Multipath Network Coding in Lossy Wireless Networks

Xinyu Zhang, Baochun Li*
Department of Electrical and Computer Engineering
University of Toronto
Email: {xzhang, bli}@eecg.toronto.edu

Abstract

Network coding has been a prominent approach to a series of problems that used to be considered intractable with traditional transmission paradigms. Recent work on network coding includes a substantial number of optimization based protocols, but mostly for wireline multicast networks. In this paper, we consider maximizing the benefits of network coding for unicast sessions in lossy wireless environments. We propose Optimized Multipath Network Coding (OMNC), a rate control and routing protocol that dramatically improves the throughput of lossy wireless networks. OMNC employs multiple paths to push coded packets to the destination, and uses the broadcast MAC to deliver packets between neighboring nodes. The coding and broadcast rate is allocated to transmitters by a distributed optimization algorithm that maximizes the advantage of path diversity while avoiding congestion. With extensive experiments on an emulation testbed, we find that OMNC achieves significant throughput improvement over traditional best path routing protocols, and existing multipath routing protocols with network coding.

1. Introduction

The prevalence of low-quality wireless links in real-world wireless networks has inspired routing protocols that are sustainable under unsatisfactory and lossy conditions [9]. Such protocols tend to follow the traditional shortest-path paradigm, with a path metric associated with the reception probabilities of wireless links. Traditional multipath routing has been proposed for the purpose of fault tolerance [21], but not for throughput improvement in lossy wireless networks, mainly due to the redundancy and route coupling problem [18].

As a major departure from the conventional store-and-forward transmission paradigm, network coding allows encoding operations on intermediate forwarders, and enables efficient algorithms to achieve the capacity of wireline multicast networks [12, 16]. Recent system implementations, such as MORE [6], augmented network coding upon existing routing protocols, and demonstrated its potential for wireless unicast. MORE allows the source node to continuously send random linearly coded packets through multiple opportunistic paths until the destination collects a sufficient number of packets for decoding. The paths are implicitly formed by nodes that can overhear packets, and that are closer to the destination than their predecessors. MORE relies on a heuristic algorithm to manage the coding and forwarding operations of intermediate nodes.

In this paper, we propose Optimized Multipath Network Coding (OMNC), an optimization based network coding protocol that controls the end-to-end transmission of coded packets in lossy wireless environment. Unlike traditional multipath routing protocols, OMNC fully utilizes the broadcast MAC that enables multiple downstream nodes to overhear packets with one single transmission attempt, and explores the capabilities of all intermediate forwarders that may contribute to the unicast session. Owning to the resilience of network coding to packet losses, OMNC guarantees unicast reliability without any retransmissions at the link level or above. With such advantages, OMNC may be applied to a range of lossy wireless networks such as unplanned wireless mesh networks and randomly deployed sensor networks, as long as the nodes are capable of performing linear encoding and decoding.

As in the MORE protocol, end-to-end transmissions in OMNC are carried by coded packet streams flowing through multiple opportunistic paths. However, OMNC is built upon an optimization framework that jointly optimizes multipath routing and rate control. In particular, OMNC matches the coding and broadcast rate of each node with its channel status, so as to avoid congestion, to fully explore the path diversity, and to reduce the generation of redundant packets. The outcome of this framework is a decentralized algorithm that can improve the end-to-end throughput for a unicast session.

*This work was supported in part by LG Electronics, and by Bell Canada through its Bell University Laboratories R&D program.

To validate the OMNC protocol, we implement and test it on a wireless emulation testbed that is designed for computationally intensive experiments like network coding. Experiments on large random networks show that OMNC can achieve a 245% throughput improvement on average over a traditional shortest-path routing scheme with high-throughput metric [9], which is significantly higher than the performance of MORE. By comparing the average queue size of each protocol, we find that OMNC avoids network congestion through its rate control mechanism, while MORE is oblivious of the channel status, thus resulting in a lower level of performance.

The remainder of this paper is organized as follows. In Sec. 2, we present a literature review of existing work, including multipath routing and network coding protocols. Sec. 3 overviews the OMNC operations and highlights the rate control algorithm that improves network throughput. In Sec. 4, we address practical issues and further optimizations on the implementation of OMNC. In Sec. 5, we evaluate the performance of OMNC, in comparison with related work. Finally, Sec. 6 concludes the paper.

2. Related work

Multipath routing has been extensively explored in multihop wireless networks. Typically, it requires an explicit path selection algorithm to identify disjoint paths, which are used in parallel to ensure fault-tolerance [10]. As for throughput, multipath routing has limited effectiveness because of the route coupling problem caused by interfering nodes [18]. Most of the existing multipath routing protocols are built atop the unicast MAC protocol. In contrast, an multipath opportunistic routing protocol [3] makes use of all relay nodes that may opportunistically receive a packet, and chooses one of the potential forwarders based on negotiations among them. Unfortunately, this requires complex interactions between intermediate forwarders.

Recent attempts on applying random linear code to wireless unicast have shown that it may achieve higher performance with less overhead than the opportunistic routing schemes [6, 17]. Among them, MORE [6] is the first practical system that combines random linear network coding with multipath opportunistic routing. MORE uses a centralized heuristic algorithm that selects potential forwarders and tells how many incoming packets they should wait before encoding a new packet. Unfortunately, this heuristic omits the possible congestion effects caused by multiple forwarders having new packets to transmit. The problem is especially pronounced when a large number of intermediate forwarders are involved in the unicast. In addition, it remains an open problem how many coded packets the source node has to transmit so as to save redundant transmissions while ensuring decodability at the destination. We address the above problems using a distributed optimization framework in OMNC. Instead of determining the number of packets, OMNC assigns the encoding and broadcast rate to each node in a decentralized manner, and seeking for optimized bandwidth usage and congestion avoidance.

Optimization based approaches to network coding have been extensively studied, but mostly confined to wireline multicast networks (see *e.g.* [7, 16]). In [17], the authors pointed out that network coding may also improve energy efficiency for wireless unicast. They proposed a min-cost problem to determine the transmission rate of each node. The results were subsequently applied to an unpublished system implementation, *i.e.*, the preliminary version of MORE [5]. However, we observe that their formulation has no rate control mechanism and does not explore path diversity well, which are critical to the performance of network coding for unicast transmissions (further details are provided in Sec. 5).

3. OMNC: highlights of the protocol

In this section, we first introduce the basic idea of OMNC, and then continue to discuss how it can be tailored to perform optimized operations leading to high unicast throughput.

3.1. OMNC: an overview

OMNC is designed for long lived unicast sessions in lossy wireless networks. In OMNC, the source node continuously generates packet streams from a group of data blocks using a random linear code (RLC). Coded packet streams flow through multiple paths towards the destination. Intermediate forwarders can refresh the packet streams by re-encoding existing packets and broadcasting the coded packets to downstream nodes. Once a sufficient number of packets accumulate at the destination, the original group of data blocks can be recovered. Thereafter, an uncoded ACK is sent back to the source (preferably using traditional best path routing), allowing it to start operating on a new group of data blocks.

Encoding and decoding algorithms. In random linear network coding, both the encoding and decoding operations can be regarded as matrix multiplication over a Galois field. Specifically, we group the source data into *generations*, and further split each generation into *data blocks*. We represent each generation as a matrix B, an $n \times m$ matrix, with rows being the n blocks of the generation, and columns the bytes (represented as integers from 0 to 255) of each data block. The encoding operation produces a linear combination of the original blocks by $X = R \cdot B$, where R is an $n \times n$ matrix composed of random coefficients in the Galois field $GF(2^8)$. The *coded blocks* (rows in the X matrix), together with the *coding coefficients* (rows in R), are packetized and flow as packet streams towards the destination.

The decoding operation at the destination node, in its simplest form, is the matrix multiplication $B = R^{-1} \cdot X$, where each row of X represents a coded block and each row of R represents the coding coefficients accomplished with it. The successful recovery of the original data blocks B requires that the matrix R be of full rank, *i.e.*, the destination must collect n independent coded blocks.

To reduce futile transmissions, an intermediate relay accepts an incoming packet only if it is independent of existing received ones, *i.e.*, it is *innovative*. The intermediate forwarders can refresh the packet streams by *re-encoding* incoming packets and *broadcasting* the resulting packets to downstream nodes. The re-encoding operation replaces the coding coefficients accomplished with the original coded packets with another set of random coefficients. The ability of re-encoding enables forwarders to avoid the severe packet redundancies in store-and-forward routing protocols, since a coded packet carries information from not only the newly coming packet, but also existing ones that were opportunistically received.

The necessity of rate control. The very nature of randomized network coding makes it possible to guarantee full reliability even under severe losses, since the probability of decoding failure approaches 0 as more and more packets accumulate at the destination [12]. However, it is nontrivial to tailor the RLC for efficient unicast, given the possible redundancy induced by linearly dependent packets, and congestion caused by neighboring nodes that interfere with each other. The key contribution of OMNC lies in its ability to manage the encoding, broadcasting and multipath routing in an optimized manner, thereby maximizing the performance of lossy wireless networks. This is mainly achieved by its rate control algorithm, which we detail below.

3.2. The OMNC optimization framework

We focus on the unicast scenario where a source node S transmit data to the destination T with the help of multiple intermediate forwarders. Assume the original network topology has undergone a decentralized *node selection* procedure so that each relay is closer to the destination T than its predecessor. Denote the resulting topology graph as $G(V, E)$, where V is the set of selected nodes involved in the unicast and E is the set of directed links.

We first set up a broadcast MAC model as an optimization constraint. For the unicast MAC, it is known that characterizing the necessary-sufficient condition for feasible MAC schedules is NP-hard [13]. A sufficient condition for feasible schedules is [15]: $\frac{f_{ij}}{C_{ij}} + \sum_{(k,l)\in I(i,j)} \frac{f_{kl}}{C_{kl}} \leq 1$, where f_{ij} is the unicast rate on link (i,j); C_{ij} is the link capacity; $I(i,j)$ is the set of links that may interfere (i,j). A necessary condition is $\frac{f_{ij}}{C_{ij}} + \sum_{(k,l)\in Q(i,j)} \frac{f_{kl}}{C_{kl}} \leq 1$, where $Q(i,j)$ is the clique in the conflict graph that involves link (i,j) [13]. In this paper, we extend the unicast MAC model to obtain the necessary condition for feasible broadcast schedules. Unlike the traditional unit-disk graph model that assumes perfect reception within transmission range, we define transmission range as the distance where packet reception probability is below a small threshold. Hence the transmission range and interference range can be considered the same (referred to as *range*). We model an ideal broadcast MAC where competing transmitters can optimally multiplex the channel without any collisions caused by exposed terminals. Without loss of generality, we assume the capacity of $\forall(i,j) \in E$ alone is the same as the MAC layer channel capacity C. Denote b_i as the rate at which node i broadcasts packets to its downstream nodes. Two transmitters compete with each other if they fall in the range of a common receiver, and a node cannot transmit and receive at the same time. Therefore, for any receiver (and possibly transmitter) $i \in V\backslash S$, we have $b_i + \sum_j b_j \leq C, (j \in N(i))$, where $N(i)$ is the set of nodes within the range of i.

In addition, we need to model how the information is broadcast along all paths in $G(V, E)$. Denote p_{ij} as the one-way reception probability of link (i,j). Consider a basic scenario where S pushes the coded packet streams to T through two paths, each containing one forwarder, denoted as u and v ($u \notin N(v)$), respectively. We observe that if u, v have different set of linearly independent packets from S, then they can generate linearly independent packets for T with high probability. Furthermore, when links are lossy, the probability for u, v to have the same set of linearly dependent packets is as low as $(p_{Su} \cdot p_{Sv})^q$, where q is the sequences of packets broadcast from S. Thus we assume u and v can independently contribute information to T. However, it is infeasible for u and v to determine whether the information is independent of existing packets received by T, and to compute the corresponding optimal broadcast rate. As an attempt to derive a distributed but not necessarily optimal solution, we adopt the following formulation instead. Denote the information flow rate on link (i,j) as x_{ij}, then the broadcast rate of i must be able to support x_{ij} even in the face of packet losses: $b_i p_{ij} \geq x_{ij}$. This means that the links with high qualities will be favored, while those that may opportunistically receive packets and contribute to the packet streams are involved as well.

Given the above models, we formulate the throughput-maximization problem as follows:

$$\textit{sUnicast:} \quad \max \quad \gamma \tag{1}$$

$$\text{subject to:} \quad \sum_j x_{ij} - \sum_j x_{ji} = \pi(i), \tag{2}$$

$$x_{ij} \geq 0, \tag{3}$$

$$b_i + \sum_j b_j \leq C, i \neq S \tag{4}$$

$$b_i p_{ij} \geq x_{ij}, i \neq T \tag{5}$$

where $i \in V$, $(i,j) \in E$, $(j,i) \in E$, and

$$\pi(i) = \begin{cases} \gamma & \text{if } i = S, \\ -\gamma & \text{if } i = T, \\ 0 & \text{otherwise.} \end{cases}$$

Here the flow conservation property (2) holds because OMNC generates a new packet only upon a newly coming packet that is innovative. A dependent packet does not contribute to the information flow and is not counted in. The above *sUnicast* problem is a linear program and its size is proportional to the number of nodes in V, and thus it can be solved in polynomial time.

We note again that the constraint (4) is only a necessary condition for collision free broadcast schedules. However, it can serve as a common ground for comparing application layer coding/routing protocols. Moreover, feasible schedules can be generated by rescaling the broadcast rate, just as in a unicast MAC [11]. Another noteworthy point is that the throughput γ in (1) may not be the actual information flow rate, due to the possible dependence of different packet streams arriving at relays and the destination. Nevertheless, the essential objective of *sUnicast* is to derive a rate allocation vector $\boldsymbol{b}$ that takes advantage of multiple opportunistic paths and takes into account the competition among neighboring nodes, rather than to compute the absolute optimal throughput value. More importantly, it translates into a practical algorithm, which performs much better than existing heuristic solutions.

3.3. A distributed rate control algorithm

We propose the following decomposition approach to solve *sUnicast*, thus obtaining a distributed rate control algorithm. Simply put, we relax the constraint (5) that entangles the broadcast rate vector $\boldsymbol{b}$ and the information rate vector $\boldsymbol{x}$, and then solve for these two variables separately in two subproblems.

First, we relax the constraint (5) with a Lagrange multiplier vector $\boldsymbol{\lambda}$ and obtain the Lagrangian function:

$$L(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{\lambda}) = \gamma + \sum\nolimits_{(i,j)\in E} \lambda_{ij}(b_i p_{ij} - x_{ij}) \quad (6)$$

According to the duality theory, the original optimization problem *sUnicast* is equivalent to the relaxed problem:

$$\min_{\boldsymbol{\lambda}} \max_{\boldsymbol{x},\boldsymbol{b}} L(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{\lambda}) \quad (7)$$

The corresponding Lagrangian multiplier problem can be solved with the subgradient method [1]:

$$\lambda_{ij}(t+1) = [\lambda_{ij}(t) - \theta(t)(b_i p_{ij} - x_{ij})]^+ \quad (8)$$

where $[\cdot]^+$ denotes the projection onto the non-negative orthant. t is the index of the iterative steps of update. $\theta(t)$ is the step size for the iteration t. Here we adopt diminishing step sizes that guarantee convergence regardless of the initial value of $\boldsymbol{\lambda}$. Specifically, $\theta(t) = \frac{A}{B+C\cdot t}$, where A, B and C are tunable parameters that regulate convergence speed.

In addition, the corresponding primal problem $\max_{\boldsymbol{x},\boldsymbol{b}} L(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{\lambda})$ can be decomposed into two separate subproblems:

SUB1: $$\max_{\boldsymbol{x}} \quad \gamma - \sum\nolimits_{(i,j)\in E} \lambda_{ij} x_{ij} \quad (9)$$

subject to constraints (2) and (3), and

SUB2: $$\max_{\boldsymbol{b}} \quad \sum\nolimits_{(i,j)\in E} \lambda_{ij} p_{ij} b_i \quad (10)$$

subject to constraint (4).

Owning to the above decomposition, we obtain a modularized optimization of two subproblems: the multipath opportunistic routing problem (*SUB1*), and the broadcast/encoding rate allocation problem (*SUB2*). These two problems are solved separately and coordinated by the Lagrange multiplier $\boldsymbol{\lambda}$.

Problem *SUB1* assumes a structure similar to the well known min-cost flow problem. However, the flow rate on each link has no upper bound (since we relaxed the constraint (5)) and the throughput γ appears in the objective function. Considering such differences, we design an alternative approach to the problem. First, we transform the original throughput maximization problem into an utility maximization problem, where the utility $U(\gamma)$ is a monotonically increasing and strictly concave function. The $\ln(\gamma)$ function is well suited for this purpose. Such a transformation can achieve the same optimal solution to $\boldsymbol{x}$ and $\boldsymbol{b}$ as the original problem. The transformed problem is:

$$\min_{\boldsymbol{x}} \quad \sum\nolimits_{(i,j)\in E} \lambda_{ij} x_{ij} - U(\gamma) \quad (11)$$

subject to constraints (2) and (3).

With respect to the vector $\boldsymbol{x}$, this is just a shortest path problem with well-established decentralized solutions. Assuming the cost of a unit flow is $p_{\min}$ (obtained by adding up the link cost λ_{ij} along the shortest path), if we send γ units of traffic through it, the total cost is $\gamma p_{\min}$. To achieve the minimal value of the objective function, *i.e.*, to take minimal cost, it is required that: $\frac{\mathrm{d}}{\mathrm{d}\gamma}[\gamma p_{\min} - U(\gamma)] = 0$, from which we obtain:

$$\gamma = U'^{-1}(p_{\min}) \quad (12)$$

Consequently, problem (11) requires us to send $U'^{-1}(p_{\min})$ units of traffic through the shortest path in each optimization iteration. By taking the Hessian of the objective function (9), it can be seen that the objective is not strictly convex, which implies the possible loss of a primal feasible solution. In view of this, we adopt the primal recovery method [20] to retain the feasibility of the primal problem:

$$x_{ij}(t) = \frac{1}{t} \sum\nolimits_{k=1}^{t} x_{ij}^k \quad (13)$$

where x_{ij}^k is the rate allocated to link (i,j) in iteration k. Note that the shortest path may change with the link cost λ_{ij} throughout the process of iterative optimization. Within each iteration, only a single shortest path is selected. However, with (13), we not only obtain a primal feasible solution, but also a multipath routing scheme that appropriately assigns rate to all links.

Table 1. Distributed Rate Control Algorithm

1. **Initialize parameters**. Set elements in $\boldsymbol{b}$, $\boldsymbol{x}$ to small positive numbers. Initialize the dual variables to 0.
2. **Solve the main framework** (7). Repeat the following steps until convergence:
3. **Solve problem SUB1:** Find the shortest path in a distributed manner, with link cost λ_{ij}. Update the information rate x_{ij} according to (12)(13).
4. **Solve problem SUB2:** for each node $i \in V$, update the primal variable b_i with (17)(18). Update the *congestion price* β_i with (15). Send the updated β_i and b_i to neighbors.
5. Update the Lagrange multiplier λ_{ij} with (8).

Next, we proceed to solve problem *SUB2* using Lagrangian relaxation. The Lagrangian form of *SUB2* is:

$$\min_{\boldsymbol{\beta}} \max_{\boldsymbol{b}} \sum_{i \in V} w_i b_i - \beta_i(b_i + \sum_j b_j - C) \quad (14)$$

where $w_i = \sum_j \lambda_{ij} p_{ij}$, $\forall (i,j) \in E$; β_i is the Lagrange multiplier, whose concrete meaning is the *congestion price* charged on node i for its violation of the channel capacity. Such congestion price can be generated by a MAC protocol itself [8]. This again justifies the practical implications of the OMNC formulation, although its scheduling constraint (4) does not perfectly model a real MAC protocol.

And again, the Lagrangian multiplier problem for *SUB2* can be solved using the subgradient method:

$$\beta_i(t+1) = \left[\beta_i + \theta(t)(b_i(t) + \sum_j b_j - C)\right]^+ \quad (15)$$

where we adopt the same step size $\theta(t)$ as in (8). The Lagrangian subproblem for (14) can be linearized as:

$$\max_{\boldsymbol{b}} \quad \sum_{i \in V} (w_i - \beta_i - \sum_j \beta_j) b_i + \beta_i C \quad (16)$$

Since this problem is linear, the Lagrange multiplier method does not necessarily generate a primal solution b_i. Thus we adopt the proximal method [2] and add a quadratic term to make it strictly convex:

$$\max_{\boldsymbol{b}} \sum_{i \in V} (w_i - \beta_i - \sum_j \beta_j) b_i - \phi ||\boldsymbol{b} - \boldsymbol{b}(t)||^2 + \beta_i C$$

Then we update b_i with:

$$b_i(t+1) = b_i(t) - \frac{w_i - \beta_i - \sum \beta_j}{2\phi} \quad (17)$$

where ϕ is an arbitrarily small positive constant that enables the above update to be arbitrarily close to the optimal value of b_i. To ensure boundedness of the iterations, we add loose lower and upper bounds to the broadcast rate b_i, *i.e.*, $0 \leq b_i \leq C$, which is consistent with the constraints in the original problem. Since the vector $\boldsymbol{b}$ is also a primal variable in the primal problem (6), we apply the primal recovery method to guarantee a primal optimal solution, in a similar way to *SUB1*:

$$b_i(t) = \frac{1}{t} \sum_{k=1}^{t} b_i^k \quad (18)$$

In summary, we describe the distributed rate control mechanism for a single unicast session in Table. 1. It is straightforward to see that the problems *SUB1* and *SUB2* have unique solutions following the above procedure. In addition, the primal recovery method ensures that the optimal dual solution of the main framework (7) converges to a primal optimal solution. Therefore, the distributed rate control algorithm is guaranteed to converge. To obtain an intuitive view of the convergence property of the algorithm, we showcase the iterative evolution of the node broadcast rate for the sample topology in Fig. 1. Here we set the channel capacity to 10^5 bytes/second and tag the reception probabilities to corresponding links. The step size is chosen as: $A = 1, B = 0.5, C = 10$. From the results, we observe that the broadcast rate converges to the optimal solution within a few rounds of iterations. For more complex topologies, the convergence speed may vary with the number of nodes and links.

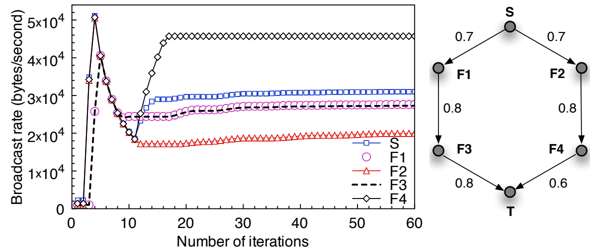

Figure 1. The convergence speed of the distributed algorithm.

4. OMNC: practical issues

Aside from the rate control mechanism, it is also necessary to optimize the design of OMNC from the following practical aspects.

Progressive decoding. A salient feature in our implementation of OMNC is the progressive decoding using *Gauss-Jordan elimination*, which keeps the decoding matrix in its *reduced row-echelon form*. *Gauss-Jordan elimination* enables the destination node to perform independence check and decoding *on-the-fly*, rather than waiting until all n independent packets in a generation are gathered and then decoded at once. A non-innovative packet will produce an all-zero row in the reduced matrix and will be discarded immediately. Once n independent packets are gathered, the left part of the reduced matrix becomes an identity matrix and the right part is exactly the original uncoded blocks from the source node. Such an implementation is important for alleviating the delay effects caused by network coding, which can be translated into throughput improvement in practice.

Accelerated network coding. To further optimize the network coding implementation, we have designed an accelerated framework for both the encoding and progressive decoding process using x86 SSE2 instructions. Instead of the traditional lookup-table approach [6], we perform the matrix multiplication on-the-fly using a loop based approach in

Rijndael's finite field. The loop based multiplication makes it possible to process two bytes of a row within one execution facilitated by the SSE2 instructions. Compared with a baseline implementation without acceleration, the coding efficiency of our framework can be 3 to 5 times higher, depending on the size of a generation and a data block.

Node Selection and Multipath Construction. Recall that a node selection procedure is needed to select potential forwarders. After node selection, the multiple opportunistic paths are constructed implicitly — all selected forwarders contribute to the unicast by re-encoding and rebroadcasting existing innovative packets, following the rate vector $\boldsymbol{b}$. Unlike the traditional multipath routing protocols [10, 21], no explicit node-joint or link-disjoint paths need to be computed.

During the node selection procedure, each node needs to compute its distance to the destination using the shortest path algorithm. Then the source node broadcasts a packet containing distance information, and the receivers are selected and continue the broadcasting if they are closer to the destination. To obtain deterministic information about the proximity, the node selection procedure uses the pseudo-broadcast proposed by Katti *et al.* [14], which ensures reliable broadcast to each neighboring node with minimal cost.

When running the shortest path algorithm, we adopt the expected transmission count (ETX) [9] as the path metric, which estimates the total number of transmissions needed to deliver a packet over a specific link, and is computed by $ETX = \frac{1}{p_{ij}}$ for link (i, j). The reception probability p_{ij} is measured by broadcasting probing packets, and taking the ratio of correctly received packets over the number that are sent [9]. OMNC is based on the presumption that the *link qualities in the target network are relatively stable* over time. Real world measurements observed that the link qualities in static wireless networks experience noticeable variations only on a daily basis [19]. Such experiments justify our extensive use of link reception probability to model the target lossy networks. In cases where link qualities change significantly, the node selection and rate allocation have to be re-initiated, which brings a certain amount of overhead. Considering the large performance gains, however, it is still more preferable than traditional routing, especially for long lived unicast sessions.

Packet and Queue Management. OMNC manages packet queues in accordance to the random linear network coding scheme. All outgoing packets are generated by re-encoding existing innovative packets, at a rate assigned by the rate control algorithm. Some intermediate nodes, especially those close to the source, may collect a whole generation of independent blocks prior to the destination. These nodes no longer accept packets from upstream nodes since all incoming packets will be non-innovative. However, they continue re-encoding packets and broadcasting them to downstreams at the specified rate, until current generation is decodable at the destination. At that time, the source node receives an ACK sent by the destination, and continues to the next generation. Either an ACK or a coded packet with a higher generation ID will dictate the intermediate nodes to discard packets belonging to the expired generation.

5. Evaluation of OMNC

Before describing our experiments, we first briefly introduce *Drift*, the emulation testbed that we use to implement and validate the OMNC protocol. *Drift* is a high performance emulation testbed that we designed for prototyping and validating application layer protocols in large-scale wireless networks. As in existing wireless emulation testbeds, application algorithms developed in *Drift* run in real-time and real operating systems. *Drift* directly employs the IP and transport layer protocol stacks in the emulation hosts, simulates the wireless PHY and MAC with specific models, and emulates wireless transmissions over a Gigabit Ethernet. The lower layer models consist of a PHY model that captures the lossy nature of the actual wireless environment, and a MAC model that captures the channel competition among neighboring nodes.

To model the opportunistic reception in a lossy wireless environment, the widely used unit-disk graph model, which assumes perfect reception within transmission range, no longer holds. Instead, we use a PHY model based on real-world traces from [4], which empirically maps link distance to the reception probability.

To model the unicast channel access, we adopt an ideal scheduling scheme in which interfering nodes (nodes within range of each other) can optimally multiplex the channel. A node cannot receive packets if it falls in the range of an interfering node. Note that the broadcast MAC in Sec. 3 is just a variant of this model. Although the ideal MAC does not model protocol details such as RTS/CTS, it provides insights for the general performance of an application-specific protocol when it is subject to MAC level competitions.

To evaluate the performance of the OMNC protocol, we have implemented it within *Drift*, together with its counterpart MORE [5,6], and the high-throughput single-path routing protocol with the ETX metric [9] (henceforth referred to as ETX routing). For the ETX routing, we assume that reliability is guaranteed by MAC layer re-transmissions, which is more efficient than the end-to-end re-transmission [17]. We now proceed to present the experimental results obtained from the *Drift* testbed.

First of all, we quantify the improvement of end-to-end throughput owning to the OMNC protocol for a single unicast session. The target topology consists of 300 randomly deployed nodes with density 6, *i.e.*, each node has on average 5 neighbors within its range (defined as the distance

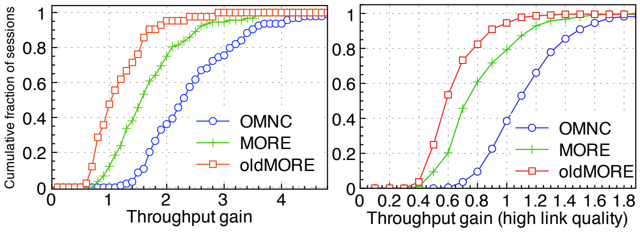

Figure 2. The distribution of throughput gains in a lossy network (left) and in a network with high link qualities (right).

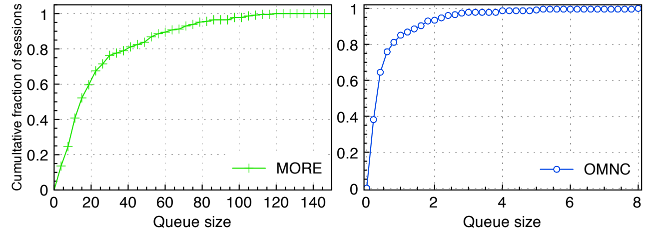

Figure 3. The distribution of time-averaged queue size in a lossy network.

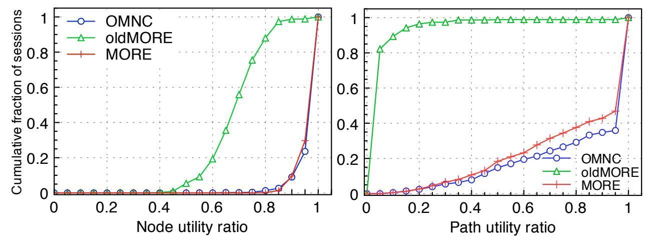

Figure 4. The distribution of node and path utility ratio in a lossy network.

where reception probability is 0.2). Most links have intermediate qualities (average reception probability is 0.58). To guarantee a fair comparison, we choose the same coding parameters for OMNC and MORE. Specifically, each generation contains 40 data blocks and each data block is of 1 KB. Both protocols share the same encoding and decoding modules, *i.e.*, the computation efficiency is identical. We adopt *throughput gain* as the metric of comparison, which is defined as the throughput of each network coding protocol divided by that of the ETX routing. The source and destination of each unicast session are randomly chosen, with a path length constraint of 4 to 10 hops. We run 300 UDP constant bit rate (CBR) sessions in total, each lasting 800 seconds. The CBR rate is set to half of the channel capacity (10^4 B/s). Throughput is calculated immediately after the source receives the "successfully decoded" ACK from the destination, and then averaged over the entire session. The resulting distribution of throughput gains is plotted in Fig. 2 (left).

As expected, OMNC has a much larger throughput gain than MORE in general. The average throughput gain of OMNC and MORE are 2.45 and 1.67, respectively. That is, OMNC can achieve 47% higher throughput than MORE. The preliminary version of MORE [5] (referred to as *oldMORE*) experiences even lower throughput gain, which is only 1.12 on average.

One additional observation is that the benefits of OMNC are best demonstrated in lossy networks, owning to its resilience to packet losses and the saved transmissions with the broadcast MAC. Fig. 2 (right) illustrates the experiment results from the same topology, but the transmission power of each node is increased such that the average reception probability rises to 0.91. In this case, the average throughput gain of OMNC is 1.12, while MORE and oldMORE actually perform worse than the ETX routing due to severe packet dependences. Nevertheless, the case where most links have intermediate qualities is more prevalent in reality, due to the severe path-loss and multipath fading typically seen in realistic wireless mesh networks [9].

Recall that OMNC jointly optimizes routing and rate control by taking into account the channel congestion status, while MORE has no rate control mechanism. As an intuitive explanation of such differences and the consequence, we monitor the channel congestion status of both protocols while they are running in the lossy topology. Specifically, we sample the broadcast queue size, take the time average, and then calculate the average queue size of each node involved in the transmission. The resulting distribution of average queue size is illustrated in Fig. 3. For most of the sessions, the per-node time-averaged queue size in OMNC is smaller than 1 (the overall average is 0.63), implying that it can match the encoding and broadcast rate of a node to its channel status. By contrast, the overall average queue size of MORE is 22. Therefore, although the heuristic in MORE tells each node how many packets it should generate, it is not aware of whether the packets can be sent out. In summary, *injecting the packet streams at a low rate may not fully utilize the channel resources, while a higher rate may cause congestion*. MORE does not address this fundamental trade-off, hence leading to the performance degradation.

The oldMORE protocol does not have any rate control mechanism, either. An additional defect is that it does not explore path diversity well. This can be illustrated by its node utility ratio (the actual number of nodes involved in the transmission divided by the total number of selected nodes), and path utility ratio (the total number of paths involved in the transmission divided by the total number of available paths after the node selection procedure), as shown in Fig. 4. The oldMORE protocol tends to prune a large number of nodes associated with low quality links, and fails to explore path diversity well, which is critical for increasing throughput. In contrast, OMNC takes advantage of all nodes that may overhear packets and contribute to the unicast, and its throughput gains are consistently higher than oldMORE. Such contrast mainly comes from the broadcast constraint (5) in OMNC, and the corresponding one in [5, 17] which favors high-quality paths. Noticeably, the new version of MORE has similar node utility ratio and path utility ratio with OMNC.

We have also observed that the actual emulated throughput of OMNC tends to be lower than the optimized throughput computed by the *sUnicast* framework, especially for the non-lossy case. This is straightforward as we noted that the constraint (4) only approximates the actual propagation of innovative flows under lossy environment (Sec. 3.2).

Regarding the convergence of the distributed rate control algorithm derived from *sUnicast*, we observe that most sessions can obtain the optimized rate vector with an acceptable number of iterations. The average number of iterations required for the experiments in Fig. 2 is 91. Beside the shortest path algorithm, the only step that needs message passing is in equation (15) and (17), where each node sends its rate and congestion price to its neighbors. And the node selection process significantly reduces the number of nodes involved in the rate control algorithm. Moreover, the rate control mechanism only has to be run *once* for each unicast, and re-initiated only if the link qualities change. Overall, the *sUnicast* algorithm can serve as a lightweight application layer protocol that improves the throughput of lossy wireless mesh networks.

6. Conclusion

In this paper, we introduced the design and implementation of the OMNC protocol and evaluated its performance. OMNC fully explores the wireless broadcast nature and path diversity, while taking advantage of network coding to adapt to the lossy environment. These salient properties are reflected in a distributed algorithm that allocates the encoding and broadcasting rate to all transmitters. With such properties, OMNC achieves significant throughput improvement over traditional routing and existing network coding protocols. As the rate control framework can be flexibly extended to other scenarios such as the multiple-unicast case, we believe OMNC marks an important step towards optimization based protocol design for network coding in unicast networks.

References

[1] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

[2] D. P. Bertsekas and J. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., 1989.

[3] S. Biswas and R. Morris. ExOR: Opportunistic Multi-hop Routing for Wireless Networks. In *Proc. of ACM SIGCOMM*, 2005.

[4] J. Camp, J. Robinson, C. Steger, and E. Knightly. Measurement Driven Deployment of a Two-tier Urban Mesh Access Network. In *Proc. of ACM MobiSys*, 2006.

[5] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. MORE: A Network Coding Approach to Opportunistic Routing. Technical Report MIT-CSAIL-TR-2006-049, MIT CSAIL, June 2006.

[6] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. In *Proc. of ACM SIGCOMM*, 2007.

[7] L. Chen, T. Ho, S. Low, M. Chiang, and J. Doyle. Optimization Based Rate Control for Multicast With Network Coding. In *Proc. of IEEE INFOCOM*, 2007.

[8] L. Chen, S. Low, M. Chiang, and J. Doyle. Joint Congestion Control and Media Access Control Design for Ad Hoc Wireless Networks. In *Proc. of IEEE INFOCOM*, 2005.

[9] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A High-throughput Path Metric for Multi-hop Wireless Routing. In *Proc. of ACM MobiCom*, 2003.

[10] P. Djukic and S. Valaee. Reliable Packet Transmissions in Multipath Routed Wireless Networks. *IEEE Transactions on Mobile Computing*, 5(5), 2006.

[11] Z. Fang and B. Bensaou. Fair Bandwidth Sharing Algorithms Based on Game Theory Frameworks for Wireless Adhoc Networks. In *Proc. of IEEE INFOCOM*, 2004.

[12] T. Ho, M. Medard, J. Shi, M. Effros, and D. Karger. On Randomized Network Coding. In *Proc. of Allerton*, 2003.

[13] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of Interference on Multi-hop Wireless Network Performance. In *Proc. of ACM MobiCom*, 2003.

[14] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft. XORs in The Air: Practical Wireless Network Coding. In *Proc. of ACM SIGCOMM*, July 2006.

[15] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Algorithmic Aspects of Capacity in Wireless Networks. *SIGMETRICS Perform. Eval. Rev.*, 33(1), 2005.

[16] Z. Li, B. Li, D. Jiang, and L. C. Lau. On Achieving Optimal Throughput With Network Coding. In *Proc. of IEEE INFOCOM*, 2005.

[17] D. Lun, M. Medard, and R. Koetter. Network Coding for Efficient Wireless Unicast. In *Proc. of IEEE International Zurich Seminar on Communications*, 2006.

[18] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. Tabrizi. On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks. In *Proc. of ACM MobiHoc*, 2000.

[19] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based Models of Delivery and Interference in Static Wireless Networks. In *Proc. of ACM SIGCOMM*, 2006.

[20] H. D. Sherali and G. Choi. Recovery of Primal Solutions When Using Subgradient Optimization Methods to Solve Lagrangian Duals of Linear Programs. In *Operations Research Letter*, volume 19, 1996.

[21] A. Tsirigos and Z. Haas. Analysis of Multipath Routing-Part I: The Effect On The Packet Delivery Ratio. *IEEE Transactions on Wireless Communications*, 3(1), Jan 2004.