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 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 network coding while avoiding congestion. With extensive experiments on an emulation testbed, we find that OMNC achieves more than two-fold throughput increase on average compared to traditional best path routing, and significant improvement over existing multipath routing protocols with network coding. The performance improvement is notable not only for one unicast session, but also when multiple concurrent unicast sessions coexist in the network.**

I. Introduction

The prevalence of low-quality links in real-world wireless mesh networks [1] has posed a fundamental challenge for mesh network designers, *i.e.*, to maintain network performance under unsatisfactory and lossy conditions. Using measured reception probability in path selection, loss-aware single-path routing protocols [2], [3] have demonstrated remarkable effectiveness in maintaining end-to-end throughput. Such protocols adopt the average link reception probability as the distance metric when running the shortest-path algorithm. Traditional multipath routing protocols [4]–[6] can be used to improve fault tolerance by delivering redundant packets through multiple paths. Opportunistic mulitpath routing further improves throughput by exploiting the wireless broadcast nature and the overhearing capability of neighboring nodes. For example, the ExOR [7] protocol allows all neighboring nodes that are closer to the destination to serve as potential forwarders, and uses delicate scheduling policies to reduce duplicate transmission. To apply its scheduling algorithm in a harsh wireless environment, however, ExOR needs nontrivial control overhead. In addition, it does not guarantee full reliability — end-to-end retransmissions are required to compensate for missing packets.

To address such deficiencies, Chachulski *et al.* [8] proposed an opportunistic multipath network coding protocol, MORE, which augmented *randomized network coding* upon opportunistic routing. MORE allows the source node to continuously send coded packets through multiple opportunistic paths until the destination collects a sufficient number of packets for decoding. With random mixing of overheard packets, each intermediate relay can generate and forward linearly independent packets with high probability, thereby avoiding the complicated negotiation that determines which packets to transmit. In addition, MORE achieves 100% packet delivery ratio, since it continues transmission until an entire chunk of file is correctly decoded by the destination.

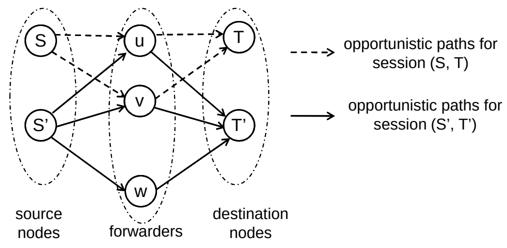
S
S'
u
v
w
T
T'
opportunistic paths for session (S, T)
opportunistic paths for session (S', T')
source nodes
forwarders
destination nodes

Fig. 1. An example scenario in multipath network coding where credit based algorithms [8] are ineffective.

Although the problem of *which packet to transmit* is circumvented by network coding, a scheduling algorithm must still be in position to determine *when to transmit*. Intuitively, relays along efficient paths should be allowed to generate and transmit coded packets more frequently, while other relays are employed opportunistically. MORE adopts a credit based heuristic algorithm that centrally prescribes for each relay how many incoming packets it should wait before generating a new coded packet. However, this approach does not account for the bandwidth resource competition among neighboring forwarders having new packets to transmit. As a result, relays in congested hotspots may generate more coded packets than the network can accommodate, rendering the credit algorithm ineffective. The problem is especially pronounced when multiple unicast sessions (*i.e.*, end-to-end flows) are running simultaneously.

As justification, consider the scenario in Fig. 1 with two concurrent unicast sessions, represented as source-destination pairs (S, T) and (S', T'). The relay nodes u and v assist both sessions, while w does not forward packets for S as the reception probability of link (S, w) is negligible. Also, since w is further away from S' than the other two relays, it will be assigned less credit according to MORE. However, in case when both sessions are heavily loaded, it is advantageous for

S' to employ w more as it causes less interference to other nodes and is less affected by the congestion. In addition to the above problem, it remains unknown how many coded packets each source node has to transmit so as to save redundant transmissions while ensuring decodability at the destination. Since a source node has an indefinite number of packets to transmit, it will inevitably congest the network without proper control of the encoding and transmission rate.

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 a lossy wireless environment. Similar to the opportunistic network coding protocol MORE, end-to-end transmissions in OMNC are carried by coded packet streams flowing through multiple paths that are formed by overhearing forwarders. However, unlike the credit algorithm in MORE, OMNC is built upon an optimization framework that jointly optimizes multipath routing and rate control. Instead of determining the number of packets, it assigns specific encoding and broadcast rate to the source and intermediate forwarders, taking into account the congestion caused by competing flows, and seeking for optimized bandwidth usage.

To derive an optimized rate for each node, we model the coding, routing and broadcast MAC constraints that are specific to a network coding based opportunistic routing protocol, with the objective of improving the end-to-end throughput for a single unicast in lossy mesh networks, and maximizing the aggregate network throughput while guaranteeing fairness when multiple unicast sessions are running. By solving the optimization framework using mathematical decomposition [9], we design a set of decentralized algorithms that match 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.

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 demonstrate that OMNC can achieve a 245% throughput improvement on average over a loss-aware single-path routing scheme with high-throughput metric [2], which is significantly higher than the performance of MORE. Besides, OMNC is always able to keep the highest aggregate network throughput when compared with existing unicast network coding protocols [8], [10], no matter how many concurrent unicast transmissions are undergoing. Our experimental results also validate the intuition that an unoptimized network coding protocol tends to produce coded packets greedily, oblivious of the channel congestion effect, hence resulting in degraded performance.

The remainder of this paper is organized as follows. In Sec. II, we present a literature review of existing work, including multipath routing and network coding protocols. Sec. III overviews the OMNC operations and highlights the rate control algorithms with diverse objectives. In Sec. IV, we address practical issues and further optimizations on the implementation of OMNC. In Sec. V, we evaluate the performance of OMNC, in comparison with related work. Finally, Sec. VI concludes the paper.

II. Related Work

Multipath routing has been extensively explored in wireless networks for the purpose of energy efficiency and fault tolerance. In traditional multipath routing, packets are duplicated [6] or erasure-coded [5] and routed through several paths in parallel. Existing research in this direction focused on optimally distributing data flows to available paths, so as to increase the probability of successful decoding at the destination node. Multipath routing has also been used to sidestep congested hotspots or alleviate long-term congestion [11] in networks with reliable links and known geographical information. All the above protocols are built atop the unicast MAC protocol and explicit path selection algorithms such as link-disjoint routing [12]. By contrast, in an opportunistic routing protocol [7], [13], [14], paths are implicitly formed: all potential forwarders that overhear a packet can contribute to the unicast, and the best relay is dynamically selected after sophisticted negotiation among them.

With opportunistic routing, each data session runs more aggressively, since more nodes are involved in the unicast. Therefore, the resource optimization problem becomes more important than in traditional multipath routing. Despite its remarkable performance advantage, the modeling and optimization of opportunistic routing remains a challenge, in contrast with the vast literature of cross-layer optimization [15], which are mostly focused on single best-path routing. In [16], optimal multipath routing is formulated as a utility maximization problem, but without considering the broadcast nature and interference constraint of wireless networks. Recent work by Zeng *et al.* [14] extended the conflict graph model to account for the opportunistic reception of broadcast packets. End-to-end throughput is optimized based on a linear programming model, which can only be centrally solved using optimization packages. In addition, they do not consider the bandwidth allocation among multiple unicast sessions, which has a large impact on the throughput performance of opportunistic routing.

Our work further differs from the above line of research in a focus on opportunistic multipath network coding. In particular , we are interested in modeling the propagation of information flows when relays are allowed to mix packets using random linear code [17]. In addition, we derive simple models for the broadcast MAC and multipath routing which, combined with the coding model, results in an optimization framework and decentralized algorithms that is amenable to practical implementation.

Optimization based approaches to network coding have been extensively studied, but mostly confined to multicast networks without interference constraints (see *e.g.* [18]). This is because the optimized flow structure in such networks can be easily mapped to actual topology, where overlapping flows of different multicast trees are directly encoded. Lun *et al.* [19] pointed out that network coding may also improve energy efficiency for unicast sessions in wireless networks. 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 [20]. However, their formulation has no rate control mechanism and does not well exploit path diversity, which are critical to the performance of network coding for unicast transmissions (further justifications are provided in Sec. V).

In [21], Radunovic *et al.* addressed resource allocation problem for network coding based on a stochastic network optimization approach. They extended the optimal backpressure algorithm for network coding based multicast [22] to the unicast case. The backpressure algorithm assumes that each intermediate forwarder has the queue size information at all downstream nodes, which is infeasible due to the difficulty of real-time feedback in a lossy wireless network. Such queuing information is not required in OMNC.

Our previous work [23] adopted an optimization approach towards multipath network coding, in order to alleviate the congestion when neighboring relays serve for a single source-destination pair. When multiple unicast sessions co-exist, the greedy nature of multipath network coding becomes a more critical problem, since each session fully utilizes all possible opportunities to maximize throughput, oblivious of the interest of other concurrent sessions. In this paper, we resolve the conflict of interest among competing flows using a more general optimization framework that maximizes the aggregate network throughput. Correspondingly, we design an OMNC algorithm for multiple unicast sessions, which is decentralized with respect to not only each session, but also to each node. We observe that OMNC achieves much higher performance than existing network coding and routing protocols, even when the max-min fairness constraint is enforced.

III. OMNC: Highlights of the Protocol

In this section, we first introduce the basic idea of a multipath network coding protocol (exemplified by MORE [8]), and then continue to present an overview of OMNC. In designing OMNC, we emphasize the formulation and design of its rate control mechanism, which can be tailored to perform optimized operations with different objectives and for different scenarios.

A. Background: Multipath Network Coding

Multipath network coding (henceforth referred to as *MNC*), first implemented in [8], is designed for long lasting unicast sessions in lossy wireless networks. In MNC, the source node first divide the original data file into trunks called *generations*. It then continuously produces packet streams from a generation using 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 accumulates 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 routing), allowing it to start operating on a new generation of data blocks.

When performing random linear network coding, both the encoding and decoding operations can be regarded as matrix multiplication over a Galois field. Specifically, each generation is split into *data blocks*. One generation can be represented 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 randomly selected 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 by the source 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.

The major task of intermediate forwarders is to refresh the packet streams, *i.e.*, to *re-encode* incoming packets and *broadcast* the resulting packets to downstream nodes. 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. To further reduce futile transmissions, an incoming packet is accepted only if it is independent of existing received ones, *i.e.*, it is *innovative*. This ensures that if a node accepts a new packet, it is also able to produce and contribute an independent coded packet to the packet streams.

B. OMNC: An Overview of the Protocol Operations

The very nature of randomized network coding enables MNC 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 [17]. However, it is a nontrivial task to tailor network coding for efficient unicast, given the possible redundancy induced by linearly dependent packets, and congestion caused by competing nodes even for the single unicast case. The key contribution of our OMNC protocol lies in its ability to manage the encoding, broadcasting and multipath routing in an optimized manner, in order to maximize the performance of lossy wireless networks. This is mainly achieved by its rate control algorithm.

Fig. 2 illustrates the flow of operation at an intermediate node that is running the OMNC rate control mechanism. Whenever a packet is overheard and buffered, it first undergoes innovative check which determines whether it is linearly independent of existing packets. If so, it will be put into an *innovative packet queue*. All outgoing packets are generated

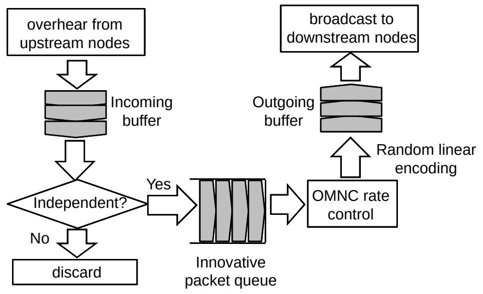

Fig. 2. The control flow of OMNC protocol at an intermediate forwarder.

by re-encoding packets in this queue, *at a rate specified 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 fresh 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 (along the shortest path), 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. The key objective of rate control is to match the encoding rate with the allowable broadcast rate, which depends on the channel congestion status in the neighborhood.

For the case of multiple unicast sessions, a relay maintains separate innovative packet queues for each session. Correspondingly, the rate control mechanism assigns specific encoding rate to each queue. Operations at the source node also follow Fig. 2, except that all incoming packets are from the original data file. Note that a credit algorithm like MORE [8] tends to cause congestion at the source side, since it can keep on incrementing its credit by obtaining packets from the data file whose size is indefinite. By contrast, OMNC constrains the coding and broadcasting rate of the source in an optimized manner, such that neighboring relays have the chance to forward the packets they have produced.

In what follows, we introduce the design of the core component in OMNC, *i.e.*, the rate control algorithm which assigns encoding rate to each transmitter in the network.

C. The Optimization Framework for A Single Unicast Session

For ease of exposition, we begin with the case for a single unicast session. Even in this case, the congestion problem is nontrivial, as multiple neighboring nodes still contend for the bandwidth resource. Before running OMNC, the source node needs to perform a shortest path algorithm with the ETX metric (exptected number of transmissions needed to deliver a packet) [2], and a *node selection* operation, which reduces the topology size by pruning those relays further away from the destination than their predecessors in terms of ETX. For instance, the relay w in Fig. 1 will be pruned by source S after node selection (More practical aspects of the node selection procedure are discussed in Sec. IV-B).

Denote the source and destination as S and T, and the resulting topology graph after node selection 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 by extending existing models for unicast MAC protocols.

Broadcast MAC model. For a unicast MAC, it is known that characterizing the necessary-sufficient condition for feasible MAC schedules is an NP-hard problem [24]. A sufficient condition for feasible schedules is [25]: $\frac{f_{ij}}{C_{ij}} + \sum_{(k,l)\in I(i,j)} \frac{f_{kl}}{C_{kl}} \le 1$, 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 $\frac{f_{ij}}{C_{ij}} + \sum_{(k,l)\in Q(i,j)} \frac{f_{kl}}{C_{kl}} \le 1$, where $Q(i, j)$ is the clique in the conflict graph that involves link (i, j) [24]. In this paper, we extend the unicast MAC model to obtain a necessary condition for feasible broadcast schedules. Different from 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 certain threshold. For simplicity, we assume the threshold is small enough, such that the transmission range and interference range can be considered the same (referred to as *range*). Beyond the range, the probability of successful reception can be ignored, and the interference is also negligible.

We model an ideal time-slotted broadcast MAC where competing transmitters can optimally multiplex the channel without any collisions. Let $B_i[t]$ be the decision variable indicating whether node i is transmitting in slot t. Then a schedule is collision free iff:

$$B_i[t] + \sum_{j\in N(i)} B_j[t] \le 1, \forall i \in V\backslash S, \tag{1}$$

i.e., in each time slot, any receiver i allows the broadcast transmission from at most one transmitter within its range. $N(i)$ denotes the set of all transmitters within its range (including itself). Note that the source node S is excluded because all nodes in V are receivers except S. Denote T as the period of a schedule, and b_i as the rate at which node i broadcasts packets to its downstream nodes, then we have:

$$b_i = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} B_i[t], \tag{2}$$

Apply (2) to (1), we obtain:

$$b_i + \sum_{j\in N(i)} b_j \le C, \forall i \in V\backslash S \tag{3}$$

where $C = \frac{1}{T}$ is the MAC layer capacity, which equals to the maximal broadcast rate of a node when no interferer presents. In consequence, for any feasible broadcast schedule, (3) must be necessarily satisfied.

We emphasize that the above model represents a necessary but insufficient condition for collision free broadcast schedules, as we transformed an integer variable $B_i[t]$ into a continuous one b_i by averaging. However, it can serve as a common ground for comparing application layer coding/routing protocols. For unicast MAC, it has been observed that rescaling the transmission rate of each link can generate feasible schedules [26]. Similar technique can be applied to the case for broadcast MAC. An alternative approach to represent a broadcast MAC is to extend the clique graph model, as in [14]. The clique model is also a necessary condition for collision free broadcast, but computing the cliques in a graph is NP-hard [24]. Hence it is not applicable to a practical optimization based algorithm like OMNC.

Coding model. In addition, we need to model how coded information is broadcast along all paths in $G(V, E)$, which is the key difference between multipath network coding and opportunistic routing [14]. Denote p_{ij} as the one-way reception probability of link (i, j). Consider a basic single-unicast scenario where source S pushes the coded packet streams to T through two paths, each containing one forwarder, denoted as u and v which cannot overhear each other (*i.e.*, $u \notin N(v)$). This is exactly the case for the session (S, T) shown in Fig. 1. 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, and p_{Su} is the reception probability of link (S, u). 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}, which is the average injection rate of innovative packets on link (i, j), 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.

Multipath opportunistic routing model. Though OMNC integrates network coding with multipath opportunistic routing, it does not violate the flow conservation constraint. At each intermediate forwarder, the outgoing rate of fresh information flows must equal the incoming rate of innovative packets. Reflected in its actual operations (Fig. 2), OMNC encodes 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. Consequently, we can derive the following flow conservation constraint to represent the routing model in OMNC:

$$\sum_{j:(i,j)\in E} x_{ij} - \sum_{j:(j,i)\in E} x_{ji} \quad = \quad \pi(i), \forall i \in V.$$

where, denoting γ as throughput, $\pi(i)$ is defined as:

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

The sUnicast problem. Given the above models, and the non-negative constraint on information flows, we formulate the throughput-maximization problem for a single unicast session as follows:

$$\begin{aligned} &\textit{sUnicast:} \qquad \max \quad \gamma, \quad \text{subject to:} \\ &\sum_{j:(i,j)\in E} x_{ij} - \sum_{j:(j,i)\in E} x_{ji} = \pi(i), \forall i \in V && (4)\\ &x_{ij} \geq 0, \forall (i,j) \in E && (5)\\ &b_i + \sum_{j\in N(i)} b_j \leq C, \forall i \in V \backslash S && (6)\\ &b_i p_{ij} \geq x_{ij}, \forall i \in V \backslash T, j \in N(i) && (7)\end{aligned}$$

One noteworthy point is that the throughput γ may be less than the actual information flow rate, due to the possible dependence of different packet streams arriving at the destination, which has been omitted in the approximated coding constraint (7). 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.

D. A Distributed Rate Control Algorithm

The above *sUnicast* problem is a linear program with size proportional to the number of nodes in V, and thus can be solved in polynomial time. Optimization decomposition [9] is well-suited for solving such a problem in a decentralized manner. Peculiar to the *sUnicast* problem, we propose the following dual-decomposition procedure [9] that leads to a distributed rate control algorithm. Simply put, we relax the constraint (7) that entangles two primal variables – the broadcast rate vector $\boldsymbol{b}$ and the information rate vector $\boldsymbol{x}$ – and then solve for these two variables separately in two subproblems. The corresponding dual problem is iteratively solved, in parallel with the primal one.

First, we relax the complicating constraint (7) with a Lagrange multiplier vector $\boldsymbol{\lambda}$ (*i.e.*, the dual variable) and obtain the Lagrangian function:

$$L(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{\lambda}) = \gamma + \sum\nolimits_{(i,j)\in E} \lambda_{ij}(b_i p_{ij} - x_{ij}) \qquad (8)$$

According to the duality theory [27], the original optimization problem *sUnicast* is equivalent to the relaxed problem:

$$\min_{\boldsymbol{\lambda}} \max_{\boldsymbol{x}, \boldsymbol{b}} L(\boldsymbol{b}, \boldsymbol{x}, \boldsymbol{\lambda}) \qquad (9)$$

The corresponding Lagrangian multiplier problem (*i.e.*, the dual problem) can be solved with the subgradient method [9]:

$$\lambda_{ij}(t+1) = [\lambda_{ij}(t) - \theta(t)(b_i p_{ij} - x_{ij})]^+ \qquad (10)$$

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:

$$\textit{SUB1:} \qquad \max_{\boldsymbol{x}} \quad \gamma - \sum\nolimits_{(i,j)\in E} \lambda_{ij} x_{ij} \tag{11}$$

subject to constraints (4) and (5), and

$$\textit{SUB2:} \qquad \max_{\boldsymbol{b}} \quad \sum\nolimits_{(i,j)\in E} \lambda_{ij} p_{ij} b_i \tag{12}$$

subject to constraint (6).

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, *i.e.*, the dual variable $\boldsymbol{\lambda}$.

Problem *SUB1* assumes a structure similar to the well known min-cost flow problem [28]. However, the flow rate on each link has no upper bound (since we relaxed the constraint (7)) and the throughput γ appears in the objective function. Considering such differences, we adopt 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:

$$\min_{\boldsymbol{x}} \quad \sum\nolimits_{(i,j)\in E} \lambda_{ij} x_{ij} - U(\gamma) \tag{13}$$

subject to constraints (4) and (5).

With respect to the vector $\boldsymbol{x}$, this is just a shortest path problem with well-established decentralized solutions [28]. 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, the first-order optimality condition [27] must hold: $\frac{\mathrm{d}}{\mathrm{d}\gamma}\left[\gamma p_{\min} - U(\gamma)\right] = 0$, from which we obtain:

$$\gamma = U'^{-1}(p_{\min}) \tag{14}$$

Consequently, problem (13) 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 (11), 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 [29] to retain the feasibility of the primal problem. Instead of applying the result from (14) directly to per-link flow rate, we take an equally-weighted average of the resulting flow rate in each iteration:

$$x_{ij}(t) = \frac{1}{t}\sum_{k=1}^{t} x_{ij}^{k} \tag{15}$$

where x_{ij}^k is the result for 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 (15), we not only obtain a primal feasible solution, but also a multipath routing scheme that assigns appropriate rate to all links.

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

$$\min_{\boldsymbol{\beta}} \ \max_{\boldsymbol{b}} \ \sum\nolimits_{i\in V} w_i b_i - \beta_i (b_i + \sum\nolimits_j b_j - C) \tag{16}$$

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 [30]. This again justifies the practical implications of the OMNC formulation, although its scheduling constraint (6) does not perfectly model a real MAC protocol.

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

$$\beta_i(t+1) = \left[\beta_i + \theta(t)(b_i(t) + \sum\nolimits_j b_j - C)\right]^+ \tag{17}$$

where we adopt the same step size $\theta(t)$ as in (10). The Lagrangian subproblem for (16) can be linearized as:

$$\max_{\boldsymbol{b}} \quad \sum\nolimits_{i\in V} (w_i - \beta_i - \sum\nolimits_j \beta_j) b_i + \beta_i C \tag{18}$$

Since this problem is linear, the Lagrange multiplier method does not necessarily generate a primal solution b_i [27]. Thus we adopt the proximal method [28] and add a quadratic term to make it strictly convex:

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

Then we update b_i with:

$$b_i(t+1) = b_i(t) - \frac{w_i - \beta_i - \sum \beta_j}{2\phi} \tag{19}$$

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.*, $0 \le b_i \le C$, which is consistent with the constraints in the original problem. Since the vector $\boldsymbol{b}$ is also a primal variable in the primal problem (8), we apply the primal recovery method to guarantee a primal optimal solution, in a similar way to (15):

$$b_i(t) = \frac{1}{t}\sum_{k=1}^{t} b_i^k \tag{20}$$

In summary, we describe the distributed rate control mechanism for a single unicast session in Table. I. 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 (9) converges to a primal optimal solution. Therefore, the distributed rate control algorithm is guaranteed to converge. A formal proof of uniqueness and

TABLE I
DISTRIBUTED RATE CONTROL ALGORITHM

```
1) Initialize parameters. Set elements in b, x to small positive numbers. Initialize the dual variables to zero.
2) For all node (link) involved in the transmission, iteratively repeat the following operations until convergence.
3) Solve the main framework (9).
    a) Solve problem SUB1. Find the shortest path in a distributed manner, with link cost $\lambda_{ij}$. Update the conceptual unicast rate $x_{ij}$ according to (14)(15).
    b) Solve problem SUB2.
        i) Update the primal variable $b_i$ with (19)(20).
        ii) Update the congestion price $\beta_i$ with (17). Send the updated $\beta_i$ and $b_i$ to all neighbors.
4) Update the Lagrange multiplier $\lambda_{ij}$ with (10).
```

convergence follows directly from the results for subgradient and decomposition method [9], [27] and is omitted in this paper. 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. 3. 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.

E. Extension to Multiple Unicast Sessions

When multiple unicast sessions are running, each runs the node selection algorithm separately, and they may share relays along the paths. Therefore, the congestion problem becomes even more critical. In this case, OMNC can assign encoding rate to each node, so as to alleviate congestion and to maximize the aggregate network throughput. Denote the session index as k,the set of concurrent sessions as K, the corresponding optimization problem is formulated as:

$$\textit{mUnicast:} \quad \max \quad \sum \gamma^k \quad \text{subject to:} \tag{21}$$

$$\sum_j x_{ij}^k - \sum_j x_{ji}^k = \pi(i,k), \tag{22}$$

$$x_{ij}^k \geq 0, \tag{23}$$

$$\sum_k b_i^k + \sum_k \sum_j b_j^k \leq C, i \neq S^k \tag{24}$$

$$b_i^k p_{ij} \geq x_{ij}^k, i \neq T^k \tag{25}$$

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

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

Unlike the single unicast case, a session index k is attached to each variable. Because of this new complication, the originally independent variables become coupled and the previous approach to the single unicast problem cannot be applied straightforwardly. Instead, we use the primal decomposition method [9]. By fixing b_i^k in the constraint (25), a primal decomposition of the *mUnicast* problem can be obtained :

$$\max \quad \sum_{k \in K} \gamma^k \tag{26}$$

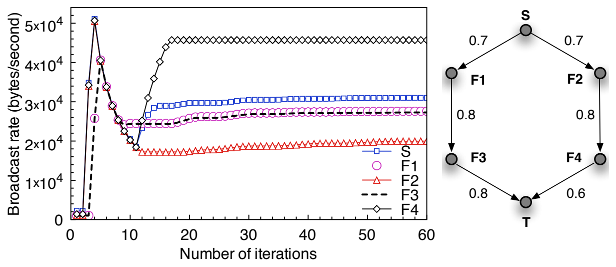

Fig. 3. The convergence speed of the distributed algorithm. Step size is chosen as: $A = 1, B = 0.5, C = 10$. F denotes forwarders. Channel capacity is 10^5 bytes/second. The reception probability is marked on each link.

subject to constraints (22), (23) and (25). This is a simple max flow problem that is decomposable with respect to k. In particular, we relax the constraint (25) and apply the subgradient method together with the primal recovery, in a similar manner to the solution of the problem *SUB1*.

With the primal decomposition, the corresponding master primal problem is:

$$\max_{\boldsymbol{b}} \quad f(\boldsymbol{b}) \tag{27}$$

subject to constraint (24), where $f(\boldsymbol{b})$ is the optimal value of the original objective as a function of the vector $\boldsymbol{b}$. Note that the analytical form of the function $f(\boldsymbol{b})$ is unavailable. However, it is convex and its subgradient equals to the Lagrange multiplier in problem (26) (for a general proof, see Sec. 6.5.3 in [27]). Therefore, we can apply the subgradient method again and update $\boldsymbol{b}$ using:

$$b_i^k(t+1) = \left[b_i^k(t) + \theta(t) v_i^k(t)\right]_{\mathcal{P}} \tag{28}$$

where v_i^k is the subgradient of the function $f(\boldsymbol{b})$ with respect to $\boldsymbol{b}$. Obviously, it is equivalent to the Lagrange multiplier associated with constraint (25) in the problem (26). $[\cdot]_{\mathcal{P}}$ denotes the projection onto the domain $\mathcal{P}$:

$$\left\{\boldsymbol{b} : \sum_{k \in K} b_i^k + \sum_{k \in K} \sum_j b_j^k \leq C, i \in V^k \backslash S^k, (i,j) \in E^k \right\} \tag{29}$$

Assume we obtain a vector $\boldsymbol{b}_0$ after the non-projected update in (28), then the projection process is equivalent to:

$$\min \quad ||\boldsymbol{b} - \boldsymbol{b}_0||^2 \tag{30}$$

$$\text{subject to} \quad \boldsymbol{A} \cdot \boldsymbol{b} \leq \boldsymbol{C} \tag{31}$$

The inequality (31) corresponds to the matrix form of the domain $\mathcal{P}$, where we stack the b_i^k to form a row vector $\boldsymbol{b}$. Then we apply Lagrangian relaxation again and obtain the Lagrangian subproblem:

$$\min_{\boldsymbol{b}} \quad \boldsymbol{b}^T\boldsymbol{b} - 2\boldsymbol{b}^T\boldsymbol{b}_0 + \sum_i \alpha_i \left(\sum_j A_{ij} b_j - C\right) \tag{32}$$

where α_i is the Lagrange multiplier and can be updated in a similar way to (17). Then each element of $\boldsymbol{b}$ can be obtained by $b_n = b_{0n} - 0.5 \sum_m \alpha_m$, where m denotes the index of the rows with non-zero A_{mi}. Note that the non-zero elements of each row in $\boldsymbol{A}$ only involve nodes in the same neighborhood, thus the problem can still be solved in a distributed manner with a few rounds of message passing. The convergence of the above algorithm follow the general convergence properties of subgradient and decomposition method [9], [27].

As the *mUnicast* problem seeks to maximize the aggregate throughput, it may be applied to admission control of multiple concurrent sessions, *i.e.*, to decide whether a newly joined session will compromise the social welfare of the entire network. However, it does not guarantee fairness among all sessions. Some sessions may be suppressed in order to maintain the total throughput. Hence we propose an alternative problem *mfUnicast* that imposes max-min fairness:

$$\textit{mfUnicast:} \qquad \max \quad \min_{k \in K} \ \gamma^k \tag{33}$$

subject to constraints (22), (23), (24) and (25). This problem is equivalent to the following linear program:

$$\max \quad \gamma \tag{34}$$

subject to $\gamma \le \gamma^k (\forall k \in K)$, plus constraints (22), (23), (24) and (25). In this paper, we only numerically solve this *mfUnicast* problem, and leave the decentralized solution to future work.

IV. OMNC: Practical Issues

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

A. Coding Operations

Progressive decoding. Implementing the random linear code in a naive way may induce heavy computation load on the wireless nodes and even result in delayed transmissions [20]. 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*. A non-innovative packet will produce an all-zero row in the reduced matrix and will be discarded immediately. On the other hand, *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. Once the destination gathers n independent packets, 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 [8], 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.

B. Node Selection and Multipath Construction

Recall that each data session needs a node selection operation to select potential forwarders. During the node selection procedure, each node computes its distance to the destination using the shortest path algorithm, using the expected transmission count (ETX) [2] as a path metric. 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. This operation continues iteratively and terminates at the destination. To obtain deterministic information about the proximity, the node selection procedure uses the pseudo-broadcast proposed by Katti *et al.* [10], which ensures reliable broadcast to each neighboring node with minimal cost.

The path metric (*i.e.*, the expected transmission count (ETX) [2]) 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 [2]. Similar to existing opportunistic routing protocols [7], 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 [31]. 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.

After node selection, the paths to the destination are constructed implicitly — all selected forwarders contribute to the unicast by re-encoding and rebroadcasting existing innovative packets, following the rate assigned to it (in the vector $\boldsymbol{b}$). Unlike the traditional multipath routing protocols [5], [11], [12], no explicit node-joint or link-disjoint paths need to be identified.

C. Packet Management

In OMNC, each packet contains the coded data block, the corresponding coding coefficients, and other protocol specific overhead. Note that the coding coefficients cause a certain amount of transmission overhead. However, the size of the coding coefficients is negligible when compared to the coded blocks. For example, in our experiment, we set the generation size n to 40 and block size m to 1000 bytes. Thus the overhead is only 4%, which is reasonable considering the overall benefits of network coding. As for the computation cost, it has been observed that the encoding, decoding and independence check takes negligible time when compared with the time scale of MAC scheduling in current wireless mesh networks [20].

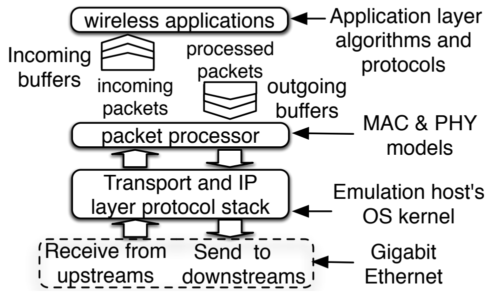

Fig. 4. The internal architecture of *Drift*.

V. Evaluation of OMNC

In this section, we first introduce *Drift*, the emulation testbed that we use to implement and validate the OMNC protocol. We then present our extensive experiments on the performance of OMNC, with respect to network throughput.

*A. Experimental Environment — The **Drift** Emulation Testbed*

Drift is a high performance emulation testbed that we designed for prototyping and validating application layer protocols in large-scale wireless networks. Compared with existing emulators (*e.g.* [32]), the main feature of *Drift* is a better trade-off between scalability and accuracy, which rests on its distributed architecture, efficient packet processing unit and analysis based lower layer models. Running in a server cluster, *Drift* is able to accommodate hundreds of nodes and several MBytes/second traffic in a single server host. More importantly, its node capacity and traffic capacity grow almost linearly with the number of emulation hosts. Such advantages enable *Drift* to operate efficiently even for those computationally intensive algorithms such as network coding.

The architectural design of *Drift* is illustrated in Fig. 4. 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. Its core component is an efficient packet processor that bridges the emulation host's kernel and the application layer algorithms, while enforcing the bandwidth constraints imposed by lower layer models. 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. We next provide more details of both models and justify their sufficiency for the purpose of evaluation. Interested readers are referred to [33] for more detailed architectural level design and evaluation of the *Drift* testbed.

PHY model. 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 [34], which empirically maps link distance to the reception probability.

MAC model. 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. III 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.

The above models in *Drift* serve as the foundation for predicting the performance trends, and ensuring a fair comparison among various application-specific protocols. Since *Drift* emulates real higher layer network stacks and applications, the functionality of the OMNC protocol (encoding, decoding, independence check, rate control, *etc.*) can be validated at real time, and on real networked machines. We now proceed to present the experimental results obtained from *Drift*.

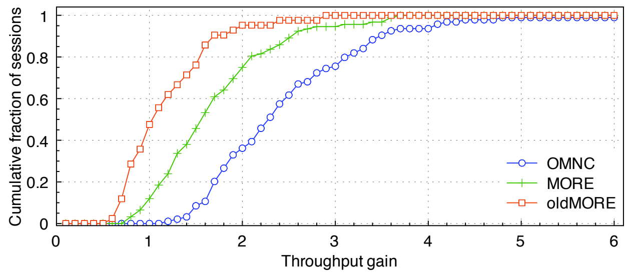

Fig. 5. The distribution of throughput gains.

B. Performance of OMNC for a Single Unicast Session

To evaluate the performance of the OMNC protocol, we have implemented it within *Drift*, together with its counterpart MORE [8], [20], and the high-throughput single-path routing protocol with the ETX metric [2] (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 [19].

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 where reception probability is 0.2). Most links have intermediate qualities (average reception probability is 0.58), in consistence with measurements of the mesh network in [1]. 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. Throughput is calculated immediately after the source receives the “successfully decoded” ACK from the destination, and then averaged over the entire session. The

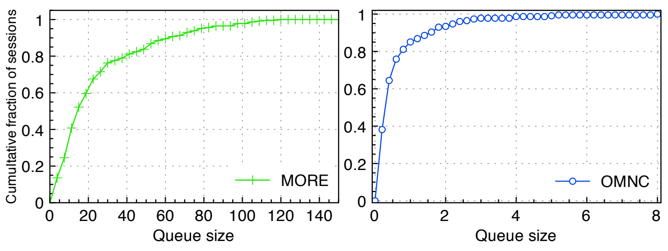

Fig. 6. The distribution of time-averaged queue size.

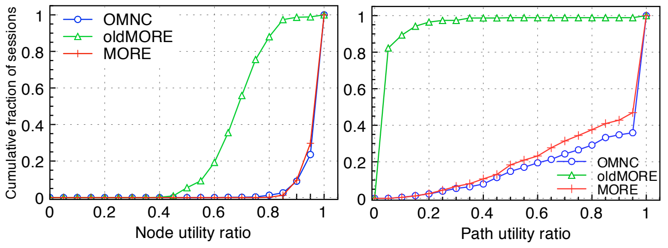

Fig. 7. The distribution of node and path utility ratio.

resulting distribution of throughput gains is plotted in Fig. 5.

As expected, OMNC has a much higher throughput gain than MORE in general. The average throughput gain of OMNC amounts to 2.45, while that of MORE is only 1.67 (this result is consistent with the evaluation in [8], which further justifies the realism of our experiments). That is, OMNC can achieve 47% higher throughput than MORE. The preliminary version of MORE [20] (referred to as *oldMORE*) experiences even lower throughput gain, which is only 1.12 on average.

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 testbed. Specifically, we sample the broadcast queue size, take the time average, and then calculate the average queue size of all nodes involved in the transmission. The resulting distribution of average queue size is illustrated in Fig. 6. 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. 7. 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 (7) in OMNC, and the corresponding one in [19], [20] which favors high-quality paths. Noticeably, the new version of MORE has similar node utility ratio and path utility ratio with OMNC.

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. In a network with high link qualities, the throughput gains are marginal due to packet dependencies. For instance, Fig. 8 illustrates the experiment results from the same topology and data sessions as in Fig. 5, but the transmission power of each node is intentionally 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 — the average gains are 0.79 and 0.63, respectively. 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 [1].

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 (6) only approximates the actual propagation of innovative flows under lossy environment (Sec. III-C). 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. 5 is 91. Beside the shortest path algorithm, the only step that needs message passing is in equation (17) and (19), 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

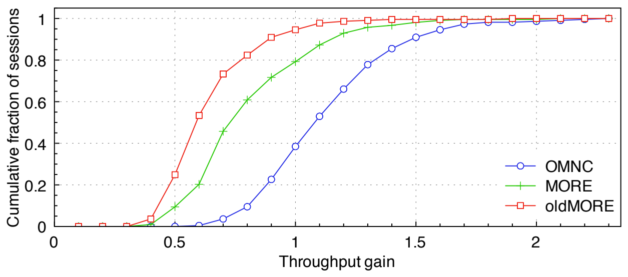

Fig. 8. The distribution of throughput gains with high link qualities.

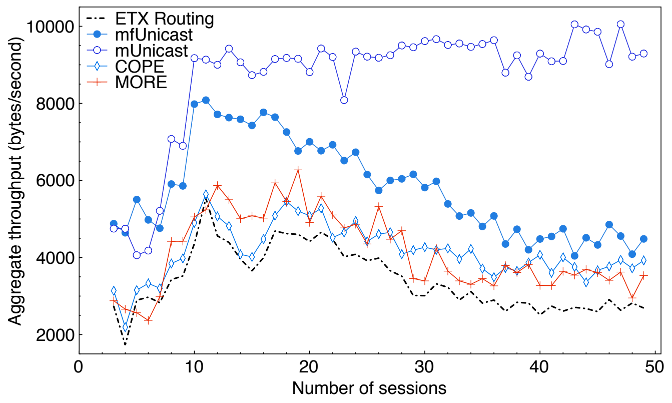

Fig. 9. The aggregate throughput of multiple unicast sessions in a lossy wireless network.

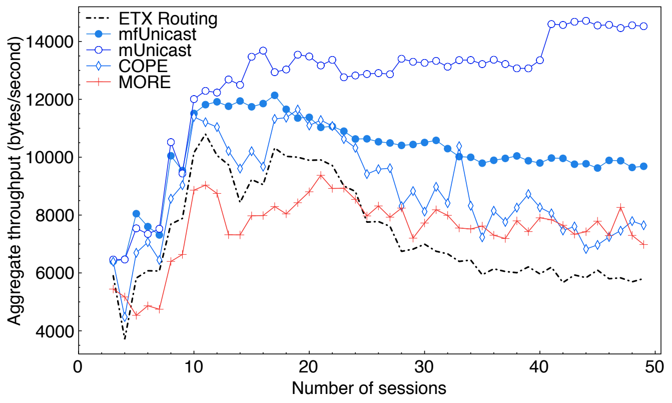

Fig. 10. The aggregate throughput of multiple unicast sessions in a network with high link qualities (average reception probability is 0.91).

throughput of lossy wireless mesh networks.

C. Performance of OMNC for Multiple Unicast Sessions

We explore the potential of OMNC for multiple unicast sessions in comparison with the ETX routing [2], the MORE [8] and COPE protocol [8]. COPE represents another family of network coding protocols, *i.e.*, inter-session network coding. It allows intermediate forwarders to opportunistically XOR incoming packets heading towards different destinations, based on prior knowledge of the decodability at the intended next-hops. The encoding nodes broadcast the coded packets to all next-hops, thereby reducing the number of transmissions compared with traditional routing. Since COPE encodes packets belonging to different source-destination pairs, *it only applies to the case with multiple-unicast sessions*. By contrast, OMNC and MORE belong to the family of intra-session network coding, which encodes packets belonging to the same source-destination pair and can be used for both single-unicast and multiple-unicast sessions.

Our experiments are performed on a 50-node random topology with node density 6 (an average of 5 neighbors per node). The reception probabilities range from 0.22 to 0.95, with an average of 0.58. The channel capacity is 10^4 bytes/second. All other parameters are the same as in the single-unicast experiment. We consider five protocols: the ETX routing, the COPE protocol [10], the MORE protocol, OMNC with the *mUnicast* and *mfUnicast* optimization, respectively. When implementing COPE, we assume that each node has precise knowledge of the overheard packets in all its neighbors, thereby giving a slightly optimistic evaluation of its performance.

Fig. 9 illustrates the aggregate network throughput as a function of traffic load (the number of randomly selected concurrent sessions). For the former four protocols, throughput increases with the traffic load in the beginning. As more sessions join, the channel becomes congested and the total throughput suffers. Since the MORE protocol does not take into account the competition between neighboring nodes and different sessions, the throughput gain suffers substantially, especially when the channel is congested. Compared with MORE, the OMNC with *mfUnicast* can achieve 2.2 times higher throughput, while OMNC with *mUnicast* retains an almost constant high throughput regardless of how many sessions are running concurrently. The COPE protocol always performs better than ETX routing, owning to the inter-session coding which reduces queue size and alleviates congestion. However, its performance dependends on the availability of coding opportunities, which are rare in lossy networks. Therefore, the performance gains are much less than OMNC.

We also consider the case when all links in the network have high qualities in Fig. 10. In this case, the MORE protocol performs worse than ETX routing when few sessions are present. This is because MORE suffers from more severe congestion as more transmitters are involved than in single-path routing. When MORE and single-path routing have similar levels of congestion, the former can achieve higher throughput owing to its extensive use of the broadcast MAC. The COPE protocol still performs better than single-path routing, and even better than in a lossy network, since it sees more coding opportunities now, and its coded packets can be more reliably broadcast to each set of downstream nodes. Again, the OMNC protocol with *mUnicast* and *mfUnicast* perform consistently better than MORE, COPE and ETX routing. But the advantage is not as significant as the lossy case. This is because OMNC results in more redundant transmissions in non-lossy environment, *i.e.*, each node tends to receive more linearly dependent data blocks from its predecessors.

Beside aggregate throughput, fairness among concurrent sessions is also an important performance metric. To evaluate the fairness of the protocols, we adopt the well-known Jain's fairness index [35], computed as $F = \frac{(\sum_{i=1}^{S} T_i)^2}{S\sum_{i=1}^{S} T_i^2}$, where T_i is the throughput of session i and S is the total number of sessions. Fig. 11 plots the fairness index of different schemes when running in the same lossy network as in Fig. 9. We observe no significant difference when only a few sessions coexist. However, as the traffic demand increases, *mfUnicast* demonstrates a much higher level of fairness than all other schemes. Without the fairness mechanism, sessions in MORE have diverse throughput, thus lower fairness. However, it still performs better than single-path protocols including COPE and ETX routing, since it takes advantage of alternative paths

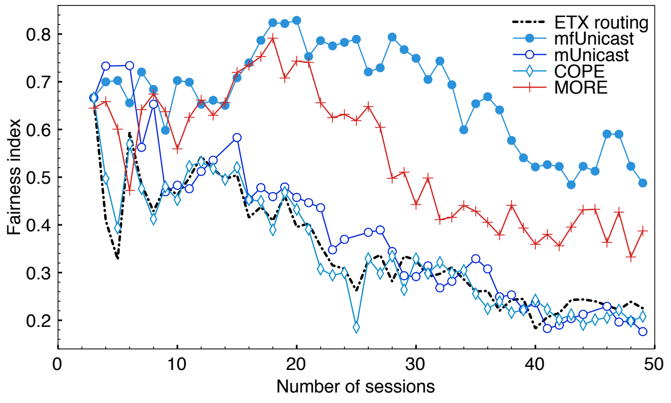

Fig. 11. The fairness of different protocols when running in a lossy wireless network.

to balance the welfare of concurrent sessions. Notably, the *mUnicast* framework results in unfairness, as it aggressively allocates bandwidth to more capable sessions, in order to maintain the total network throughput.

VI. 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 environments. These salient properties are reflected in a set of distributed algorithms that allocate the encoding and broadcasting rate to all transmitters. With such properties, OMNC achieves significant throughput improvement over traditional routing and existing network coding protocols, for both the single-unicast and multiple-unicast scenarios. As the rate control framework can be flexibly extended, we believe OMNC marks an important step towards optimization based protocol design for network coding in unicast networks.

References

[1] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and Evaluation of An Unplanned 802.11b Mesh Network," in *Proc. of ACM MobiCom*, 2005.
[2] 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.
[3] C. Koksal and H. Balakrishnan, "Quality-Aware Routing Metrics for Time-Varying Wireless Mesh Networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 11, 2006.
[4] A. Tsirigos and Z. Haas, "Analysis of Multipath Routing-Part I: The Effect On The Packet Delivery Ratio," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, Jan 2004.
[5] P. Djukic and S. Valaee, "Reliable Packet Transmissions in Multipath Routed Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, 2006.
[6] Q. Dong, S. Banerjee, M. Adler, and A. Misra, "Minimum Energy Reliable Paths Using Unreliable Wireless Links," in *Proc. of ACM MobiHoc*, 2005.
[7] S. Biswas and R. Morris, "ExOR: Opportunistic Multi-hop Routing for Wireless Networks," in *Proc. of ACM SIGCOMM*, 2005.
[8] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," in *Proc. of ACM SIGCOMM*, 2007.
[9] D.Palomar and M. Chiang, "A Tutorial on Decomposition Methods for Network Utility Maximization," in *IEEE JSAC*, Aug 2006.
[10] 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.
[11] L. Popa, C. Raiciu, I. Stoica, and D. S. Rosenblum, "Reducing Congestion Effects in Wireless Networks by Multipath Routing," in *Proc. of IEEE ICNP*, 2006.
[12] A. Srinivas and E. Modiano, "Minimum Energy Disjoint Path Routing in Wireless Ad-hoc Networks," in *Proc. of ACM MobiCom*, 2003.
[13] C. Cetinkaya and E. Knightly, "Opportunistic Traffic Scheduling Over Multiple Network Paths," in *Proc. of IEEE INFOCOM*, 2004.
[14] K. Zeng, W. Lou, and H. Zhai, "On End-to-End Throughput of Opportunistic Routing in Multirate and Multihop Wireless Networks," in *Proc. of IEEE INFOCOM*, 2008.
[15] X. Lin, N. B. Shroff, and R. Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," *IEEE Journal on Sel. Areas in Comm.*, vol. 24, no. 8, 2006.
[16] X. Lin and N. B. Shroff, "Utility Maximization for Communication Networks with Multi-path Routing," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, May 2006.
[17] T. Ho, M. Medard, J. Shi, M. Effros, and D. Karger, "On Randomized Network Coding," in *Proc. of Allerton*, 2003.
[18] 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.
[19] D. Lun, M. Medard, and R. Koetter, "Network Coding for Efficient Wireless Unicast," in *Proc. of IEEE International Zurich Seminar on Communications*, 2006.
[20] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "MORE: A Network Coding Approach to Opportunistic Routing," MIT CSAIL, Tech. Rep. MIT-CSAIL-TR-2006-049, June 2006.
[21] B. Radunovic, C. Gkantsidis, P. Key, and P. Rodriguez, "An Optimization Framework for Opportunistic Multipath Routing in Wireless Mesh Networks," in *Proc. of IEEE INFOCOM*, 2008.
[22] T. Ho and H. Viswanathan, "Dynamic Algorithms for Multicast With Intra-session Network Coding," in *Proc. of Allerton Conference*, 2005.
[23] X. Zhang and B. Li, "Optimized Multipath Network Coding in Lossy Wireless Networks," in *Proc. of IEEE ICDCS*, 2008.
[24] 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.
[25] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Algorithmic Aspects of Capacity in Wireless Networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, 2005.
[26] Z. Fang and B. Bensaou, "Fair Bandwidth Sharing Algorithms Based on Game Theory Frameworks for Wireless Ad-hoc Networks," in *Proc. of IEEE INFOCOM*, 2004.
[27] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003.
[28] D. P. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., 1989.
[29] 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*, vol. 19, 1996.
[30] 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.
[31] 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.
[32] P. Mahadevan, A. Rodriguez, D. Becker, and A. Vahdat, "MobiNet: A Scalable Emulation Infrastructure for Ad-hoc and Wireless Networks," in *WiTMeMo*, 2005.
[33] X. Zhang and B. Li, "*Drift*: A Highly Condensed Emulation Framework for Mobile Nodes in Server Clusters," in *Proc. of IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS),*, 2008.
[34] 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.
[35] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure of Fairness and Discrimination For Resource Allocation in Shared Computer Systems," DEC Research, Tech. Rep. TR-301, September 1984.