O3: Optimized Overlay-based Opportunistic Routing

Mi Kyung Han   Apurv Bhartia   Lili Qiu   Eric Rozner
The University of Texas at Austin
{hanmi2,apurvb,lili,erozner}@cs.utexas.edu

Abstract — Opportunistic routing achieves significant performance gain under lossy wireless links. In this paper, we develop a novel approach that exploits inter-flow network coding in opportunistic routing. A unique feature of our design is that it systematically optimizes end-to-end performance (*e.g.*, total throughput). A key challenge to achieve this goal is a strong tension between opportunistic routing and inter-flow network coding: to achieve high reliability, opportunistic routing uses intra-flow coding to spread information across multiple nodes; this reduces the information reaching an individual node, which in turn reduces inter-flow coding opportunity. To address this challenge, we decouple opportunistic routing and inter-flow network coding by proposing a novel framework where an overlay network performs overlay routing and inter-flow coding without worrying about packet losses, while an underlay network uses optimized opportunistic routing and rate limiting to provide efficient and reliable overlay links for the overlay network to take advantage of. Based on this framework, we develop the first optimization algorithm to jointly optimize opportunistic routes, rate limits, inter-flow and intra-flow coding. We then develop a practical opportunistic routing protocol (*O3*) based on the optimization results. Using Qualnet simulation, we study the individual and aggregate benefits of opportunistic routing, inter-flow coding, and rate limits. Our results show that (i) rate limiting significantly improves the performance of all routing protocols, (ii) opportunistic routing is beneficial under high loss rates, whereas inter-flow coding is more effective under low loss rates, and (iii) *O3* significantly out-performs state-of-the-art routing protocols by simultaneously leveraging optimized opportunistic routing, inter-flow coding, and rate limits.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms

Algorithms, Performance

Keywords

Opportunistic routing, wireless mesh networks, network coding

1. INTRODUCTION

Motivation: Providing efficient and reliable wireless communication is important because wireless losses are common. Opportunistic routing effectively combats wireless losses by taking advantage of the broadcast nature of the wireless medium (*e.g.*, [1, 2, 17]). An interesting question, which we explore in this paper, is whether opportunistic routing can benefit from inter-flow network coding, which has been successfully applied to single path routing in wireless mesh networks (*e.g.*, COPE [8]). We address this question by developing a theoretical optimization framework and designing a practical protocol to achieve the gain.

As a motivating example, consider the topology in Figure 1 with two bi-directional flows between A and D. In *traditional single path routing*, the expected number of transmissions to deliver one packet over each hop is 2 due to the 50% loss rates, and altogether 8 transmissions are required to deliver one packet for each of the two flows.

In *opportunistic routing*, a flow source uses either B or C to forward traffic (instead of only B or C). Therefore a packet makes progress if it reaches either forwarding node. This probability is 75%, assuming independent link loss, which is common in many real networks [19, 20, 25]. So on average it takes only 1.33 transmissions to move a packet over the first hop (*i.e.*, to either of the intermediate nodes) and 2 transmissions to move the packet from the intermediate node to the destination. Therefore, altogether 6.66 transmissions are required to successfully deliver both packets.

The performance of *inter-flow coding* [8] depends on whether there is an inter-flow coding opportunity. If the two flows use the same intermediate node as the forwarder, which is the best case, then it takes 6 transmissions to successfully deliver both packets (*i.e.*, 2 transmissions to deliver one packet over the first hop in both flows as in single path routing, and 2 transmissions for the intermediate node to deliver the packets to both A and D by XOR-ing them). In this case, node A can extract the packet it needs by XOR-ing its own packet with the one received from the forwarder. So can node D. When the two flows use different forwarders, there is no inter-flow coding opportunity and it takes 8 transmissions to deliver one packet for each flow as in the traditional single-path routing.

We propose to exploit inter-flow network coding in opportunistic routing. Not only does it take only 1.33 transmissions to move a packet across the first hop by using opportunistic routing, but also an intermediate node can XOR packets from the two flows whenever possible. In the *best case* (*i.e.*, the intermediate nodes can XOR all packets), the intermediate nodes only need 2 transmissions to deliver packets for both flows by XOR-ing them, which results in 4.66 transmissions in total to deliver one packet for each of the two flows. This yields a gain of 72% over single path routing, 43% over opportunistic routing alone, and 29% over inter-flow coding alone. The *worst case* (*i.e.*, intermediate nodes cannot XOR any packets),

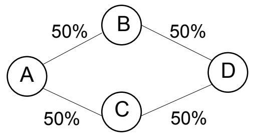


Figure 1: Leveraging inter-flow network coding in opportunistic routing.

which rarely occurs, reverts to opportunistic routing and requires 6.66 transmissions, out-performing single path routing and (worst-case) inter-flow network coding by 20%.

Challenges: The above example demonstrates the potential benefit of inter-flow network coding in opportunistic routing. However, harnessing this gain in practice poses significant challenges. There exists a *strong tension* between opportunistic routing and inter-flow coding. Opportunistic routing spreads information across multiple nodes. As the information reaching an individual node decreases, the inter-flow coding opportunity decreases because (i) the node itself receives less traffic and has more limited coding choices, and (ii) its next-hops receive less traffic, making it harder to decode. Therefore it is challenging to simultaneously leverage opportunistic forwarding to combat wireless losses and exploit inter-flow coding to reduce traffic.

Our approach: To decouple the strong interactions between opportunistic routing and inter-flow coding, we propose a novel framework to jointly optimize opportunistic routing, rate limiting, and intra- and inter-flow coding. We introduce a novel abstraction by making a wireless network consist of an overlay and underlay, where overlay nodes perform inter-flow coding aware overlay routing without worrying about packet losses and underlay nodes perform intra-flow coding based opportunistic routing without worrying about inter-flow coding.

- *Overlay network:* We designate a subset of nodes as overlay nodes and create an overlay network using them. Each traffic demand is routed over one or more overlay paths. Nodes on the overlay path perform overlay forwarding. They may also use inter-flow network coding to reduce the amount of overlay traffic generated and use inter-flow network decoding to extract the original content. For example, given two packets, one from flow $f1$ and the other from $f2$, whose overlay paths are $o_1 - o_2 - o_3$ and $o_3 - o_2 - o_1$ respectively, node o_2 may XOR the two packets and transmit the inter-coded packet. Nodes o_1 and o_3 perform inter-flow decoding to extract the packets they want. Overlay links are considered reliable so that we can focus on optimizing overlay routes, overlay rate limits, and inter-flow coding without worrying about packet losses.

- *Underlay network:* An overlay link may be mapped to one or more physical links in the underlay network. The underlay network provides efficient and reliable overlay links by using opportunistic routing to spread information across multiple forwarders and letting them cooperatively forward the traffic. To prevent fine-grained coordination, each forwarder independently generates random linear combinations of traffic from the same flow at an appropriate rate so that the destination can extract the original data after receiving enough linearly independent packets. Overlay traffic imposed on each overlay link (whether inter-flow coded or not) is considered as a virtual flow to the underlay network. The goal of an underlay network is to jointly optimize opportunistic routing and rate limiting of the virtual flows without worrying about inter-flow coding.

- *Relationship between the two:* Optimized overlay routing uses efficient overlay routes and inter-flow network coding to reduce the virtual traffic demands imposed on the underlay network. Meanwhile, optimized underlay routing provides efficient and reliable overlay links that the overlay network can take advantage of. The reason that inter-flow coding is put at the overlay network is that the optimization of inter-flow coding is much simpler without packet losses, while opportunistic routing targets packet losses and is naturally to be placed at the underlay network, which involves lossy physical links.

Based on this framework, we formulate the problem of optimizing end-to-end user performance as a linear program (LP) that optimizes total network throughput (or other linear functions) while satisfying: (i) flow conservation constraints for the overlay network, (ii) flow conservation and opportunistic constraints for the underlay network, (iii) constraints that map the traffic demands from the overlay network to the underlay network, and (iv) interference constraints. We then translate the optimization results into practical routing configurations and design an optimized overlay-based opportunistic routing protocol (*O3*) to harness the gains in practice.

We implement *O3* in Qualnet along with (i) shortest path routing (SPP), (ii) SPP with rate limiting, (iii) COPE [8], a state-of-the-art inter-flow coding based routing protocol, (iv) COPE with rate limiting, (v) MORE [2], a state-of-the-art opportunistic routing protocol, and (vi) optimized opportunistic routing, which is also called *O3*-Intra since it is *O3* without inter-flow coding. Using Qualnet simulations, we study the benefits of inter-flow coding, opportunistic routing and rate limiting, and find that (i) rate limiting is important to all routing protocols, (ii) the effectiveness of opportunistic routing increases with loss rates, but the effectiveness of inter-flow coding decreases with loss rates, (iii) *O3* significantly out-performs all the other protocols by simultaneously harnessing the gains of opportunistic routing, inter-flow coding, and rate limiting.

Our main contributions are as follows:

- A novel framework based on the concept of an overlay network to effectively decouple the strong inter-dependency between opportunistic routing and inter-flow network coding.

- The first theoretical formulation that jointly optimizes inter-flow coding, opportunistic routing, and rate limiting.

- A practical routing protocol that realizes the optimized opportunistic routes with inter-flow coding and rate limiting.

- Extensive evaluation to show the effectiveness of *O3* and the individual benefits of inter-flow coding, opportunistic routing, and rate limiting.

Paper outline: The rest of the paper is organized as follows. In Section 2, we survey related work. We give an overview of our approach in Section 3. We present a theoretical formulation of the optimization problem in Section 4. We describe how to use the optimization framework to drive the design of inter-flow aware opportunistic routing in Section 5, and present a practical routing protocol in Section 6. We describe our evaluation methodology and performance results in Section 7. We conclude in Section 8.

2. RELATED WORK

Our work is related to both opportunistic routing and inter-flow network coding, which we review below.

Opportunistic routing protocols: ExOR [1] is a seminal opportunistic routing protocol. In ExOR, a sender broadcasts a batch of packets with a list of nodes that can potentially forward these packets. In order to maximize the progress of each transmission, the forwarding nodes relay data packets in the order of their proximity to the destination. The proximity is quantified using the ETX metric [4], which reflects the expected number of transmissions required to deliver a packet from the sender to the destination. ExOR imposes strict timing constraints and coordination among the forwarders to avoid redundant transmissions.

Since then, several other opportunistic routing protocols, such as [2, 12, 16, 17, 32, 36], have been proposed. In particular, MORE [2] applies intra-flow network coding to opportunistic routing to avoid fine-grained cooperation among the forwarders and achieves significant improvement over ExOR. However, the performance of MORE degrades as the number of flows increases due to its lack of rate limiting, as shown in Section 7.

A few other studies (*e.g.*, [18, 23, 29, 30, 35]) propose optimization frameworks for opportunistic routing. Our work differs from these works in that (i) our optimization framework jointly optimizes inter-flow network coding and opportunistic routing, and (ii) the primary focus of the above works is theoretical analysis, whereas our work goes beyond theoretical analysis and develops a practical routing protocol.

Inter-flow network coding: COPE [8] develops a practical inter-flow network coding scheme for unicast in multi-hop wireless networks. There are many follow-up works that enhance COPE. For example, [5, 13, 28] develop techniques to select routes that create more coding opportunities, [3, 27] jointly optimize network coding and scheduling, [10] picks the modulation rate that takes into account both coding gain and data rate, and [33] proposes a technique to XOR packets that use different modulation schemes. Our work is built upon [28]. Different from [28], we select coding-aware opportunistic routes (instead of coding-aware traditional deterministic routes) to achieve high efficiency in the presence of wireless losses.

Researchers have mainly focused on applying inter-flow network coding to single path routing, where the routes are known before packet transmissions. Harnessing the benefit of inter-flow coding in opportunistic routing is more challenging due to uncertainty in the final routes being selected. There have been a few preliminary attempts that try to exploit inter-flow coding in opportunistic routing, as evidenced by a few short papers [11, 31, 34]. They focus only on one aspect of the routing design – among multiple nodes that receive the data, which one to pick to actually forward the data. They use EXOR-style opportunistic routing, and impose strict forwarding order, which requires significant co-ordination and limits spatial reuse. Only [22] considers the use of intra-flow coding as in MORE to avoid duplicates without coordination. However, it recognizes the significant challenges of applying inter-flow coding to general opportunistic routing, so it only supports opportunistic receptions over a single path. This significantly reduces efficiency under lossy links (*e.g.*, it behaves the same as COPE in the example in Figure 1 and requires 6 transmissions in the best case). Moreover, it does not develop a routing protocol and only uses numerical estimation of the number of transmissions based on the assumptions of a 1-packet flow with perfect acknowledgements, which make comparison hard.

In short, the existing works have four major limitations. First, they use pre-existing opportunistic routing protocols to route their data and do not select their opportunistic routes in an inter-flow coding-aware manner. Second, these heuristics try to reduce the number of transmissions but do not directly optimize end-to-end performance. The number of transmissions has been shown to have limited predictive power on end-to-end performance [8, 15]. In particular, COPE [8] shows that even in a simple 3-hop topology the coding gain (*i.e.*, the reduction in the number of transmissions) is very different from the MAC gain (*i.e.*, the improvement in throughput). Third, in order to limit the overhead of opportunistic routing, they restrict forwarding node selection, which limits the inter-flow coding opportunities. Fourth, their evaluation either uses toy topologies ([11, 34]) or compares only with COPE ([31]), and understanding the benefits of various protocols in general settings remains an open question.

Summary: It remains an open problem how to jointly optimize opportunistic routing, inter-flow coding, and rate limiting for end-to-end user performance. To solve this problem, it is necessary to develop a systematic framework that captures the effects of all these components on network performance. We develop the first optimization framework to jointly optimize opportunistic routing, inter-flow coding, and rate limiting, and design an opportunistic routing protocol based on it. In addition, we use extensive evaluation to compare a diverse set of routing schemes, and examine the individual and overall benefits of opportunistic routing, rate limiting, and inter-flow coding. Moreover, compared with previous works [11, 22, 31, 34], which report an average gain of 15-30% over COPE in random topologies, *O3* provides much higher gain over COPE. The higher performance gain demonstrates the effectiveness of our joint optimization framework.

3. OVERVIEW

O3 operates in the following three steps: (i) selecting overlay nodes and overlay paths (Section 5.1), (ii) mapping each overlay link into one or more physical links (Section 5.1), (iii) jointly optimizing overlay and underlay routing, rate limiting, and inter-flow coding based on the traffic demands, overlay network, and the mapping between the overlay and underlay networks (Section 4). The output specifies (i) how fast each source should generate traffic, (ii) how overlay nodes should forward the traffic (*e.g.*, what is the overlay path used, which nodes perform inter-flow coding, and at what rate), and (iii) how underlay nodes should opportunistically forward the traffic (*e.g.*, how many broadcast transmissions to make upon receiving traffic from its neighbor).

In Section 4, we first present an optimization framework for (iii), which takes overlay paths and mappings between overlay and underlay networks as input and outputs the optimized overlay routing, underlay routing, rate limits, and inter-flow network coding. The output is optimal when the input enumerates all possible overlay paths and maps each overlay link to the entire underlay network (*i.e.*, lets each overlay link use any underlay link for potential routing). However, this optimization problem may incur significant computation cost due to a large number of optimization variables. In Section 5, we describe our approach to improve scalability. It reduces the size of the optimization problem by selectively choosing overlay paths and mapping each overlay link to a small subset of underlay links.

Before delving into the details of each step, let us first go through a simple example shown in Figure 1, which has two flows in opposite directions. Suppose we select nodes A and D as overlay nodes; meanwhile we choose AD as an overlay path for flow 1 and choose DA as an overlay path for flow 2. Then in the overlay network, node A sends to node D via the overlay path AD, and node D sends to node A via the overlay path DA. There is no inter-flow coding since there is no intermediate overlay node in this case. If the overlay link AD is mapped to the entire physical network as an underlay, the underlay network is responsible for sending traffic for flow f1 from node A to node D using opportunistic routing on the entire underlay network. Similarly, if the overlay link DA uses the entire physical network as the underlay, then the corresponding underlay network is responsible for sending flow f2 from node D to node A using opportunistic routing. So essentially each underlay network tries to carry the traffic imposed by the corresponding overlay link l

from $src(l)$ to $dest(l)$, where $src()$ and $dest()$ denote the source and destination of the link, respectively.

Alternatively, we may select nodes A, B, C, D as overlay nodes, and choose AD, ABD, ACD as overlay paths for flow 1 and DA, DBA, DCA as overlay paths for flow 2. Then in the overlay network, node A splits its traffic across the three overlay paths according to the optimization output. So does node D. Node B may XOR flow f1's traffic sent on ABD with flow 2's traffic sent on DBA, and the fraction of inter-flow coded traffic is determined by the optimization output. Similarly for node C. As before, the underlay network is responsible for opportunistically routing all the traffic imposed by the corresponding overlay link, where the imposed traffic can be either inter-flow coded or not.

4. PROBLEM FORMULATION

Based on the overlay framework, we derive a linear program (LP) that consists of the following four components: (i) flow conservation constraints on the overlay network that can use inter-flow coding, (ii) flow conservation and opportunistic constraints on the underlay network that uses intra-flow coding based opportunistic routing, (iii) constraints mapping traffic demands from the overlay network to the underlay network, and (iv) interference constraints to prevent interfering links from being active simultaneously. The key challenge in this formulation is to accurately capture virtual flows and physical flows and interactions between the overlay and underlay networks. Below we describe the formulation in detail.

4.1 Optimization Objective

Our framework is general and can optimize any linear function. We focus on the most common metric: maximizing total throughput, namely $\sum_{k \in D} \sum_{P \in PS^k} f^k(P)$, where D is the set of traffic demands, $f^k(P)$ is the k-th flow's throughput over path P, and PS^k is the set of paths used by the k-th flow. Alternatively, we can support (i) maximizing a linear approximation of proportional fairness, defined as $\sum_{k \in D} log(\sum_{P \in PS^k} f^k(P))$, which strikes a good balance between fairness and throughput [24], (ii) maximizing the fraction of demand that is served from each flow, denoted as α, where $\alpha \cdot D_k$ is the lower bound of throughput for the k-th flow, or (iii) maximizing total revenue if the revenue is a linear function of throughput.

4.2 Overlay Network Constraints

The route on an overlay network must satisfy flow conservation. We derive the flow conservation constraints by applying coding-aware optimization for single path routing, as described in [28]. The main difference from traditional flow conservation is inter-flow coding allows an intermediate node to deliver different information to different neighbors using the same transmission. Therefore we need to classify traffic into native (*i.e.*, without inter-flow coding) and inter-coded, and derive the constraints based on the traffic type. Specifically, let $z_i^k(P)$ denote the amount of native traffic transmitted by node i for flow k over path P. Let $x_i(e1, e2, n)$ denote the amount of traffic received from link $e1$ as native traffic and transmitted by node i over link $e2$ as inter-flow coded, and $x_i(e1, e2, c)$ denote the amount of traffic received from link $e1$ as inter-flow coded traffic and transmitted by node i over link $e2$ as inter-flow coded traffic. We call $(e1, e2, n)$ and $(e1, e2, c)$ *coding structures*. Let CS denote the set of coding structures in the network. We have the following flow conservation constraints under inter-flow network coding, where $e1$ is node i's incoming link and $e2$ is node i's outgoing link.

- $\sum_{(e1,e2,n) \in CS} x_i(CS) \leq \sum_{k \in D} \sum_{e1e2 \in P, P \in PS^k} z_{t(e1)}^k(P)$. This says that the transit traffic participating in coding as *native-received* at node i is bounded by the total native traffic received from $t(e1)$, which is the transmitter of link $e1$.
- $\sum_{(e1,e2,c) \in CS} x_i(CS) \leq \sum_{k \in D} \sum_{e1e2 \in P, P \in PS^k} [f^k(P) - z_{t(e1)}^k(P)]$. This reflects that the total traffic that participates in coding as *coded-received* at node i is bounded by the amount of traffic received as coded at node i.
- $\sum_{k \in D} \sum_{e1e2 \in P, P \in PS^k} f^k(P) = \sum_{k \in D} \sum_{e1e2 \in P, P \in PS^k} z_i^k(P) + \sum_{(e1,e2,n) \in CS} x_i(CS) + \sum_{(e1,e2,c) \in CS} x_i(CS)$. This indicates the total traffic received from link $e1$ and transmitted over link $e2$ by node i must be one of the three types of traffic: (i) traffic going out as native, (ii) traffic participating in coding as native-received, and (iii) traffic participating in coding as coded-received.
- $z_{src(k)}^k(P) = f^k(P)$, where $P \in PS^k$. This indicates a flow source $src(k)$ transmits all traffic as native over every path.
- $z_i^k(P) \leq f^k(P)$, where $i \in P - \{src(k), dst(k)\}$ and $P \in PS^k$. This indicates that the amount of native traffic transmitted by a transit node is bounded by the total traffic on the path P.

4.3 Underlay Network Constraints

The goal of the underlay network is to use opportunistic routing to efficiently and reliably route the traffic demands imposed by the overlay network. The traffic includes either an original flow f or inter-flow coded traffic between multiple flows. For convenience, we denote either original or inter-flow coded traffic as *physical flow* pf. Then every combination of overlay link vl and physical flow pf is considered as a *virtual flow*, denoted by (vl, pf). For example, consider 3 physical flows in the overlay network: $f1, f2, f1 + f2$. The virtual traffic demands on the underlay network are $< o_i - o_j, f1 >, < o_i - o_j, f2 >, < o_i - o_j, f1 + f2 >$, where $o_i - o_j$ denotes any overlay link. Let $src(vl)$ and $dest(vl)$ denote the source and destination of the overlay link vl. The underlay network uses optimized opportunistic routing to efficiently route the physical flow pf from $src(vl)$ to $dest(vl)$.

Underlay flow conservation constraints: To ensure valid opportunistic routes on the underlay, we first derive flow conservation constraints for each virtual flow. Different from traditional flow conservation, the flow conservation constraints of the underlay only apply to the amount of information (*i.e.*, non-redundant useful data) instead of traffic due to packet losses. Let $Y(vl, pf, i, j)$ denote the information transmitted from node i to node j for the virtual flow (vl, pf).

- $Y(vl, pf, k, src(vl)) = 0$ for any node k. This enforces no incoming information to $src(vl)$ for a virtual flow (vl, pf) since $src(vl)$ is the source of the virtual flow.
- $Y(vl, pf, dest(vl), k) = 0$ for any node k. This enforces no outgoing information from $dest(vl)$ for a virtual flow (vl, pf) since $dest(vl)$ is the destination of the virtual flow.
- For any transit node $i \neq src(vl)$ and $i \neq dest(vl)$, $\sum_{k \in in(i)} Y(vl, pf, k, i) \geq \sum_{j \in out(i)} Y(vl, pf, i, j)$, where $in(i)$ and $out(i)$ denote node i's incoming and outgoing neighbors, respectively. It ensures that the incoming information to node i is no less than the outgoing information from i.
- $\sum_k Y(vl, pf, src(vl), k) \leq NR(vl, pf)$. This denotes that the amount of information successfully delivered from $src(vl)$ to node k is bounded by the virtual flow's traffic demand, denoted as $NR(vl, pf)$.

Underlay opportunistic constraints: Next we capture the relationships between the amount of traffic and the amount of information delivered on the underlay network. We formulate these relationships using the following opportunistic constraints, where the first one captures the relationships for a given virtual flow while the next two constraints capture the relationships for a physical flow that spans multiple overlay links from the same overlay source. The latter constraints are necessary because we allow an overlay source to broadcast traffic over multiple overlay links simultaneously and let all its downstream nodes derive information from the same transmission. Therefore we need to ensure the total information derived across all overlay links and across all downstream nodes does not exceed the amount of successfully received traffic.

- Virtual flow opportunistic constraint: $S(i,\mathcal{N}(i))T(vl,pf,i) \geq \sum_{k\in\mathcal{N}(i)} Y(vl,pf,i,k)$, where $\mathcal{N}(i)$ denotes a subset of i's neighbors, $S(i,\mathcal{N}(i))$ is the probability of successfully delivering traffic to any node in $\mathcal{N}(i)$, and $T(vl,pf,i)$ is the amount of traffic transmitted from node i on overlay link vl for flow pf. This constraint indicates for any virtual flow (vl,pf) the total traffic successfully delivered to at least one neighbor in $\mathcal{N}(i)$ should be no less than the total amount of non-overlapping information delivered to $\mathcal{N}(i)$. When i has many (say, K) neighbors, enumerating $\mathcal{N}(i)$, all subsets of neighbors, is costly. For scalability, when $K>3$, we enumerate the neighbor sets of size 1, size 2, and the one containing all i's neighbors (*i.e.*, enumerate only $O(K^2)$ instead of $O(2^K)$ neighbor sets).

- Physical flow opportunistic constraint 1: $S(i,k)MaxT(pf,i) \geq \sum_{(i,*)\in vl} Y(vl,pf,i,k)$, where $MaxT(pf,i)$ is the total overlay traffic node i sends for physical flow pf over all overlay links. Due to the broadcast nature of overlay traffic (*i.e.*, an overlay node can use a single transmission to send a packet along multiple overlay paths by including all the overlay paths in the packet header), $MaxT(pf,i) = max_{vl} T(vl,pf,i)$. These constraints together enforce that total information delivered from i to k over all virtual links is bounded by the total traffic successfully delivered from node i to k for the physical flow pf.

- Physical flow opportunistic constraint 2: This constraint further ensures that the total amount of information delivered to a subset of i's neighbors, denoted as $\mathcal{N}(i)$, over all virtual links is bounded by the product of i's traffic and the probability of successfully delivering to at least one neighbor in $\mathcal{N}(i)$:
$$S(i,\mathcal{N}(i))MaxT(pf,i) \geq \sum_{k\in\mathcal{N}(i)} \sum_{(i,*)\in vl} Y(vl,pf,i,k).$$
To improve scalability, we use the same enumeration procedure as in constructing the virtual flow opportunistic constraints (*i.e.*, enumerating the neighbor sets of size 1, size 2, and the one containing all i's neighbors when $K>3$).

4.4 Constraints Relating Overlay to Underlay

To relate the overlay to the underlay network, we derive the following constraints. The first two constraints relate the traffic demands of the virtual flow with the overlay traffic, and the last constraint ensures the virtual flow is serviced by the underlay network:

- $NR(vl,pf) = \sum_{vl\in P} z^k_{src(vl)}(P)$, where $pf = (native,k)$. This reflects that the traffic demand for a native flow (vl,pf), denoted as NR, is equal to the amount of native traffic flow k sent over the virtual link vl.

- $NR(vl,pf) = \sum_{vl\in P} x_{src(vl)}(CS)$, where CS is the coding structure and $pf = (coded, CS)$. This indicates that the traffic demand for a coded virtual flow (vl,pf) is equal to the coded traffic sent using the same coding structure.

- $NR(vl,pf) = \sum_{k\in in(dest(vl))} Y(vl,pf,k,dest(vl))$, which indicates that the traffic demand for the virtual flow (vl,pf) is honored by the underlay network, *i.e.*, the traffic demand $NR(vl,pf)$ is successfully delivered to $dest(vl)$.

4.5 Interference Constraints

Finally, we impose interference constraints for the traffic sent on the physical network, since this is the actual traffic transmitted. Based on the network topology, we construct a broadcast conflict graph. Specifically, two transmitters are considered to have conflict if either of the following conditions holds: (i) the two transmitters are within carrier sense range of each other, or (ii) one receiver is within the interference range of the other transmitter. We then find independent sets in the conflict graph and derive the following interference constraints that indicate the total activity time of a node is no more than the sum of activity time of all the independent sets that the node belongs to.

- Let MT_i denote the total traffic from node i. If node i is an overlay node, we have $MT_i = \sum_{pf} max_{vl} T(vl,pf,i)$; otherwise we have $MT_i = \sum_{pf}\sum_{vl} T(vl,pf,i)$. The reason for such a distinction is that the overlay node uses the broadcast nature of the wireless medium to transmit over multiple overlay links simultaneously by including these overlay links in its packet header. In comparison, underlay nodes forward for a specific overlay link and thus an underlay node needs to separately forward for each overlay link included in the received packet's header.

- For every node i, $MT_i \leq Cap_i \sum_{k\in I_i} \lambda_k$, where Cap_i is node i's broadcast data rate, I_i denotes the independent sets that node i belongs to, and λ_k denotes the activity time of independent set k. This constraint enforces the total traffic sent by any node is bounded by the sum of the activity time of the independent sets that the node belongs to scaled by the wireless capacity.

- $\sum_k \lambda_k \leq 1$ because only one independent set can be active at a time.

5. USING OPTIMIZATION FRAMEWORK

In this section, we describe how to obtain the inputs required by the optimization and how to translate the optimization results into routing configurations.

5.1 Obtaining Inputs

Our optimization algorithm requires the following inputs: network topology, traffic demands, overlay paths, and mapping from overlay to underlay network resources. The network topology can be obtained easily through periodic measurements. As reported in [6, 14], wireless traffic exhibits temporal stability, and we can estimate current traffic demands based on previous demands. Thus, here we focus on the latter two inputs. Our optimization framework in Section 4 is flexible and can easily take inputs generated by other overlay path selection and overlay-to-underlay mapping algorithms.

Selecting overlay nodes: One way to select overlay nodes is to let every physical node serve as an overlay node. This leads to the best performance at the cost of higher computation time, since the computation cost increases with the number of overlay nodes. Therefore we want to limit the number of overlay nodes. Since only intermediate overlay nodes perform inter-flow coding, our goal is to select overlay nodes with high coding opportunities.

To achieve this goal, for each flow f_k we order the nodes on its forwarding list according to the coding opportunities. We estimate the upper-bound of the coding opportunities as given by

$$min(T(f_k,i), \sum_{j \in D}(min(T(f_k,i),T(f_j,i)))), \quad (1)$$

where $f_k \neq f_j$ and $T(f_k,i)$ is total traffic transmitted by node i for flow f_k. Equation (1) is derived based on the fact that the rate of inter-flow traffic between two flows is bounded by the minimum rate of these two flows. Therefore, $min(T(f_k,i),T(f_j,i))$ gives an upper-bound on the amount of traffic that can be inter-flow coded between f_k and f_j, and Equation (1) gives an upper-bound of total traffic that can be inter-flow coded between f_k and all the other flows. For every flow, we pick the top three nodes from the sorted list as the overlay nodes. When the upper-bound is the same, we use the amount of traffic sent in either direction to break ties.

Selecting overlay paths: After selecting overlay nodes, we then generate overlay paths for each flow. Each flow contains at least one overlay path directly from the source to the destination, and this overlay path is mapped to the entire underlay network to ensure the solution is no worse than opportunistic routing alone, which is a special case of *O3*. If this is the only overlay path between the source and destination, *O3* becomes opportunistic routing alone, since this overlay path does not involve an intermediate node and there is no inter-flow coding.

To leverage inter-flow coding, a flow may contain other overlay paths going through one or more intermediate nodes. For each flow, we identify the overlay nodes (selected in the previous step) that are on the flow's forwarding list, which includes the flow source and destination. We enumerate all possible overlay paths involving these nodes, where their order on the overlay path is based on their ETX [4] (*i.e.*, the number of required transmissions to deliver a packet) to the destination.

Mapping overlay network to underlay network resources: The goal of this step is to map each overlay link to one or more physical links. Only the physical links, to which the overlay link is mapped to, can potentially be used as part of an opportunistic route; but whether these physical links actually participate in opportunistic routing and how much traffic they each route depend on the optimization result of the problem formulated in Section 4.

One possible mapping is to let each overlay link span all physical nodes and links. To enhance scalability, we treat an overlay link $o1-o2$ as a virtual traffic demand and use MORE to select nodes and links to be included in the underlay network. Specifically, we find the forwarding list for this virtual flow from $o1$ to $o2$ using MORE. The overlay link then uses all nodes on the forwarding list as underlay nodes, and uses physical links between these nodes as underlay links. The intuition behind this mapping is that links on the opportunistic routes are most useful for forwarding traffic from $o1$ to $o2$.

5.2 Executing Optimization

The optimization can run at a central location that distributes the optimization results to all nodes. The amount of information to distribute is small compared to data traffic. Specifically, the input includes traffic demands, link loss rates, and the conflict graph, which are $O(F)$, $O(E)$, $O(E^2)$, respectively, where F is the number of flows and E is the number of physical links. Among these three terms, $O(E^2)$ is a dominating term, so the input requires $O(E^2)$. The output includes overlay and underlay credits, which are $O(ON \cdot F \cdot P)$ and $O(N \cdot D \cdot F \cdot OE) + O(N \cdot D \cdot OE^2)$, respectively, where ON is the number of overlay nodes, N is the number of physical nodes, P is the number of overlay paths, D is the number of physical neighbors, and OE is the number of overlay links. Therefore we can tradeoff between the wireless performance and the size of information to be exchanged by controlling the number of overlay nodes and links. Moreover, only non-zero credits need to be exchanged. From our experience, a large majority of credits are zero so the actual information to be exchanged is well below the above worst case (*e.g.*, only a few KB for a 25-node network in our simulation).

Instead of centralized computation, the computation can be done in a distributed fashion, similar to link-state protocols like OSPF [21], where every node implements the same algorithm over the same data to arrive at the same results. The amount of link state information is very small. The optimization is executed either periodically or upon changes in network topology or traffic conditions. The computation time is reasonable (*e.g.*, around 3.6 seconds for 4 flows in 25-node random networks used in our evaluation). To further enhance scalability, when the inputs change slightly, we can leverage incremental LP solvers, such as lp_solve_inc [7], to take advantage of incremental changes in the linear constraints and more efficiently derive a solution to the new LP rather than solving it from scratch.

In addition to optimization based on the global information, as part of our future work, we are interested in applying decomposition techniques developed for distributed convex optimization (*e.g.*, [9]) to solve the optimization based on decentralized information to further enhance the scalability.

5.3 From LP Output to Routing Configurations

The optimization results specify the desired sending rates for both inter- and intra-flow coded traffic. A flow source i transmits at the rate of $max_{vl}T(vl,pf,i)$ for its flow pf. An intermediate node uses a credit-based scheme to enforce its forwarding strategy according to the derived $T(vl,pf,i)$, where pf can be either inter-flow or intra-flow coded. Specifically, underlay nodes do not care about inter-flow coding and simply forward traffic pf according to $T(vl,pf,i)$. Overlay nodes perform inter- and intra- encoding and decoding as specified in Section 6.1. Since the exact rate of sending inter-flow coded traffic at an overlay node depends on traffic dynamics and is hard to enforce, we convert the desired traffic rates into intra-flow credits and use inter-flow coding whenever an opportunity arises. Note that our credit computation is different from [2] due to significant difference in the two routing protocols (*e.g.*, *O3* needs to compute overlay and underlay credits, whereas [2] has only one type of credit). Below we specify credit computation for underlay and overlay nodes based on the LP output.

The credit is defined as the number of transmissions that should be generated for every received packet. Upon receiving a packet, a node increments its credit. When this credit becomes greater than or equal to 1, it generates a transmission and then decrements its credit by 1. This process is repeated until its credit goes below 1. Based on this credit definition, we can compute the credit as the total desired sending rate divided by the total receiving rate. Credit information is then stored as the following tuples: $(f,P,i,credit)$ for overlay nodes, $(f,vl,prev(i),i,credit)$ for underlay nodes' intra-coding credits, and $(vl1,vl2,prev(i),i,credit)$ for underlay nodes' inter-coding credits, where f is the flow id, P is the overlay path id, i is the node id, vl is the overlay link id, $prev(i)$ is the previous hop of node i in the underlay network, $vl1-vl2$ is the overlay segment and $prev(i)-i$ is an underlay link that is responsible for forwarding traffic for the overlay segment.

We first compute underlay credits. Upon receiving a transmission from node j, underlay node i increments its credit by $C \times R$, where C reflects the fraction of useful information contained in each transmission from node j, and R reflects the amount of redundancy node i should include to compensate for loss to its forwarders. Therefore, we have $C = Y(vl,pf,j,i)/(TC(vl,pf,j) * (1-loss(j,i))$, where its numerator is the amount of information received and its denominator is the amount of traffic received, and their ratio gives the amount of information contained in a received packet. $R = T(vl, pf, i)/\sum_k Y(vl, pf, i, k)$. R's numerator is the desirable sending rate, its denominator is the total information successfully delivered to its forwarders k's, and their ratio indicates how much traffic to generate in order to deliver one-packet worth information to i's forwarders.

Next we compute overlay credits. Upon receiving intra-flow coded traffic, an overlay node i increments its credit for a given path P and flow f by

$$T(vl, f, i) * \frac{z_i^{pf}(P)}{\sum_{P_i: vl \in P_i} z_i^{pf}(P_i)},$$

where the second term in the product is how much fraction of native traffic node i received along virtual link vl is for path P, and the product indicates the total amount of native traffic received over vl for path P. Upon receiving inter-flow coded traffic, an overlay node increments its credit associated with the intra-flow involved by $(z_{src(pf)}^{pf}(P) - z_i^{pf}(P)) * NSR(i, vl)$, where the first term in the product indicates how much inter-flow coded information is at node i and $NSR(i, vl)$ is the expected number of transmissions required to successfully deliver a packet to one of i's forwarders and can be computed as

$$1.0/(1.0 - \prod_{\forall k \in fwd(i)} loss(i,k)),$$

assuming independent packet losses at different nodes, where $fwd(i)$ denotes node i's forwarding list. For example, if node i is an overlay forwarder for $f1$, upon receiving $f1 + f2$, it increments $f1$'s credit as described above.

6. PROTOCOL SPECIFICATION

We now describe how to achieve a practical routing protocol, *O3*, based on the optimization results. In this section, we first present the algorithm to perform joint inter-flow and intra-flow encoding and decoding, and then describe the behaviors of flow sources, destinations, and forwarders.

6.1 Packet Coding Algorithm

We use random linear coding to code packets within the same flow and use XOR to code packets across flows. In our implementation, we inter-code up to 2 flows, the common case for inter-coding. Our coding algorithm is general and can code more flows at a higher computational cost. Below we present the detailed algorithms.

Encoding: To code intra-flow data, a flow source $src(f)$ divides user traffic into batches, as in MORE. Each batch has K packets, where K is a tunable parameter to trade-off between batching overhead and delay. When the MAC is ready for transmission, $src(f)$ or its forwarder, generates a random linear combination of all packets it has from the current batch and broadcasts this packet. We refer to such a coded packet as an *intra-coded* packet. To code inter-flow packets from two batches, denoted as $(f1, b1)$ and $(f2, b2)$, a node first generates an intra-coded packet P_1 using a random linear combination of all packets in $(f1, b1)$, and similarly generates packet P_2 from $(f2, b2)$. Then it XORs packets P_1 and P_2 to create an *inter-coded* packet.

Decoding: Each incoming packet yields a linear constraint. If the incoming packet is intra-coded from batch $(f1, b1)$ with batch size of $K1$, the constraint involves $K1$ variables in $(f1, b1)$. If the incoming packet is inter-coding of $(f1, b1)$ and $(f2, b2)$, whose batch sizes are $K1$ and $K2$, respectively, this inter-coded packet gives one constraint involving $K1 + K2$ variables for these two batches.

The goal of intra-flow decoding is to recover the original packets from the batch. For the batch size of $K1$, a node can use Gaussian Elimination to decode the entire batch when it has $K1$ innovative (*i.e.,* linearly independent) packets.

The goal of inter-flow decoding is to extract intra-coded packets, which in turn can be used to extract the original packets from the batch. For example, if a node has everything from $(f1, b1)$, then reception of an innovative inter-coded packet with $(f1, b1) + (f2, b2)$ (*i.e.,* linearly independent of the other inter-coded packets) allows us to extract one intra-coded packet for $(f2, b2)$ using Gaussian Elimination. More generally, if the inter-coding matrix has rank r, then we can use Gaussian Elimination to extract $max(r - K1, 0)$ intra-coded packets for batch $(f2, b2)$, and extract $max(r - K2, 0)$ intra-coded packets for batch $(f1, b1)$.

To support intra- and inter- decoding, a node maintains intra- and inter-coding matrices, which store the coefficients used in all the innovative packets. The main design issue in the decoding algorithm is how to handle interactions between the intra-coding and inter-coding matrices. To simplify the encoding and decoding processes, we maintain all the information in the intra-coding matrix if there is no inter-coding matrix involving the batch; otherwise we keep information in both intra-coding and inter-coding matrices. We extract intra-coding constraints from the inter-coding matrices whenever possible and add it to the corresponding intra-coding matrix. Specifically, when a node receives a packet, it uses the packet header to determine whether it is intra-coded or inter-coded. An intra-coded packet should be added to the intra-coding matrix involving the batch to which the packet belongs, as well as to the inter-coding matrix, if the batch is involved in inter-coding. An inter-coded packet is first added to the inter-coding matrix, from which we extract an intra-coding constraint if its rank is large enough (*i.e.,* exceeding either $K1$ or $K2$). If the packet is the first inter-coded packet for the batch pair, we (i) create an inter-coding matrix, (ii) copy the intra-coding matrices to the inter-coding matrix if one or more exist (so that the inter-coding matrix maintains all the intra-flow information obtained so far), and (iii) add the new packet to the inter-coding matrix.

To reduce storage cost, we identify active batches as described in Section 6.2 and store coding matrices only for the active batches. The intra-coding matrix can be removed immediately when the corresponding batch becomes inactive, while the inter-coding matrix can be removed only when both batches in the matrix become inactive. In our evaluation, storage per node is 300 KB for 16 flows in a 25-node random topology spanning 1000x1000 m^2, which is easily affordable for today's hardware.

6.2 Flow Sources and Destinations

A flow source, $src(f)$, *never* performs inter-flow encoding and only generates intra-coded packets at the rate computed by the LP. Each packet generated by the $src(f)$ or intermediate forwarders includes all the overlay paths that the packet may traverse and the current overlay link associated with each overlay path. Further, to facilitate the decoding of any inter-coded packets in the future, $src(f)$ saves the intra-coded packet it transmits in its buffer until the corresponding batch becomes inactive.

In MORE, $src(f)$ continues transmitting packets from the current batch until it receives an ACK for the batch. This incurs significant stop-and-wait overhead. To reduce such overhead, a large batch size K would be beneficial. However, to effectively support inter-flow coding, we prefer a small batch size, since a node can start extracting a new intra-coded packet only when the rank of the inter-coding matrix exceeds K. The larger the value of K, the lower the inter-flow coding opportunity. To efficiently support a smaller batch size, we allow a flow source to send multiple batches before receiving an ACK. The destination generates an ACK either when an entire batch is received or when a threshold number of new packets are received since the last ACK. The ACK contains *(min-active-batch-id, active, status),* where *min-active-batch-id* is the id of the smallest active batch, *active* is a bit map where $active[i] = 1$ indicates batch i is active and has not been ACKed, and status is an array indicating the number of innovative packets received by the destination for each active batch. The source uses this information to schedule transmissions from different batches in a FIFO order, and the forwarders use the information to remove inactive batches.

6.3 Forwarders

In this subsection, we describe two major tasks of a forwarder: (i) processing a received packet and (ii) generating and transmitting a packet when the medium is available.

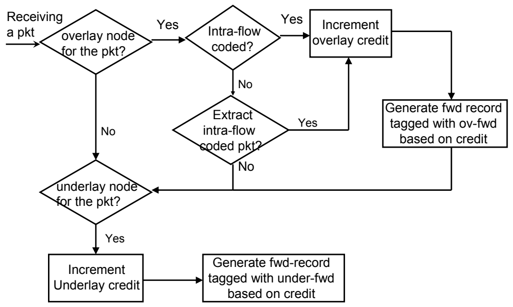

Receiving a pkt
overlay node for the pkt?
Yes
Intra-flow coded?
Yes
Increment overlay credit
No
Extract intra-flow coded pkt?
Yes
Generate fwd record tagged with ov-fwd based on credit
No
No
underlay node for the pkt?
Yes
Increment Underlay credit
Generate fwd-record tagged with under-fwd based on credit

Figure 2: Steps involved in processing a received packet at an intermediate node.

6.3.1 Process a received packet

Determine whether to perform overlay and/or underlay forwarding: As shown in Figure 2, upon receiving a packet, a node first checks if it is an overlay and underlay forwarder for this packet. It does so by inspecting the set of overlay paths/links included in the packet. A node is an overlay forwarder for an *intra-flow* coded packet if it is on at least one of the overlay paths in the packet header, and is an overlay forwarder for an *inter-flow* coded packet if it is an overlay forwarder for either $f1$ or $f2$. In either case, it invokes overlay forwarding operation. A node then checks if it is an underlay forwarder for this packet in a similar way. If it is, it invokes underlay forwarding operation. Note that it is possible for a node to perform both overlay forwarding and underlay forwarding for the same packet. If a node is neither an overlay nor underlay forwarder for a packet, it simply drops the packet.

Overlay node operation: An overlay forwarder is responsible for forwarding traffic to the next overlay node along the overlay path and performing inter-flow encoding and decoding whenever necessary. For an overlay node, if it receives an intra-flow coded packet, it looks up its credit table computed based on the optimization results as described in Section 5.3 to determine how many packets to send. Instead of generating actual packets for transmission, it generates forwarding records (one for each packet to be sent out), where the record specifies the flow, overlay path(s), batch, and the forwarding mode of the packet (*e.g.*, whether *ov-fwd* or *under-fwd*). The actual packets are not generated until the medium becomes available for transmission. Delaying packet generation until transmission allows us to make up-to-date intra-coding and inter-coding decisions.

It then tags the generated records with *ov-fwd* to indicate they are eligible for inter-flow coding, and inserts them into the queue, which will result in packet generation and transmission when the medium becomes available.

If it receives an inter-flow coded packet $P(f1,b1,f2,b2)$, it first checks whether it can extract an intra-coded packet of $(f1,b1)$ or $(f2,b2)$. If so, this reduces to the case of receiving an intra-coded packet. Otherwise, it inserts the coding coefficient into the corresponding inter-coding matrix and waits for future extraction of an intra-coded packet. This wait time is bounded by a threshold, after which the packet is garbage collected.

Underlay node operation: The goal of an underlay forwarder is to forward traffic for the current overlay link using opportunistic routing. It looks up its corresponding credit increment table as computed in Section 5.3, generates forwarding records according to the credit, tags each record with *under-fwd* (to prevent them from performing inter-flow coding), and inserts them into the output queue. Note that the processing is similar for inter-flow and intra-flow coded packets. The only difference is that a different credit table is consulted to determine the number of forwarding records to generate.

Generating forwarding records: To handle multiple outstanding batches and multiple overlay paths per flow, a forwarder not only maintains a flow credit $flow_{cr}(f,p)$ for each combination of flow f and overlay path p but also maintains a batch credit $batch_{cr}(b,f,p)$, where b is the batch id. $flow_{cr}$ determines the transmission rate for a given flow over a given overlay path, and $batch_{cr}$ determines the transmission rate for a specific batch. When receiving a packet, we update all the flow and batch credits that match f,b and $\forall p \in$ *ov-set*. A node generates a record for flow f as long as $max_p(flow_{cr}(f,p)) \geq 1$. The record includes all overlay paths p_i with $flow_{cr}(f,p_i) \geq 1$. To handle multiple batches, a record is generated from the batch with the largest batch credit over all overlay paths, *i.e.*, the largest $\sum_{p \in OS} batch_{cr}(b,f,p)$, where OS is the set of overlay paths that the current packet should be sent along. After constructing the record, an overlay forwarder tags it with *ov-fwd* whereas an underlay forwarder tags it with *under-fwd*. In both cases, the forwarder inserts the generated record to the FIFO queue, decrements the corresponding $flow_{cr}$ and $batch_{cr}$ values by 1, and continues generating new forwarding records until the $flow_{cr}$ drops below 1.

6.3.2 Transmit when medium becomes available

When the medium is available, the node dequeues forwarding records from its queue and generates a corresponding packet for transmission. More specifically, the forwarder dequeues the first forwarding record *(f1,b1,ov-set1)* from its queue and if the record is tagged with *under-fwd*, it generates a random linear combination of all packets corresponding to flow $f1$ and batch $b1$ and transmits it. If the record is tagged with *ov-fwd*, which indicates it is eligible for inter-flow coding, it searches for another record from its queue *(f2,b2,ov-set2)* that can be inter-flow coded with the first packet. If a match is found, the node dequeues *P(f2,b2,ov-set2)*, inter-flow codes the two packets, and includes (*ov-set1*, *ov-set2*) in the packet header to indicate the packet should be forwarded along the paths in *ov-set1* and *ov-set2*. Then it broadcasts the resulting inter-flow coded packet. If no match is found, the nodes generate an intra-coded packet from flow $f1$ and batch $b1$, includes *ov-set1* as the overlay path, and sends it out.

To check if two packets can be inter-flow coded, we examine the positive x_i (defined in Section 4.2) values from the LP to determine the combinations of overlay nodes and overlay paths that are involved in inter-flow coding. We store these positive values in a lookup table at each node. Two packets $P1$ and $P2$, containing the set of overlay paths *ov-set1* and *ov-set2*, respectively, can be inter-flow coded if and only if for each $ov1 \in$ *ov-set1* and $ov2 \in$ *ov-set2*, there exists an entry in the lookup table indicating we can inter-code $ov1$ and $ov2$.

To enhance inter-flow coding opportunity, we introduce two queues

Q_{inter} and Q_{intra}, where packets from Q_{intra} are usually sent out as intra-flow coded, and packets from Q_{inter} are sent out as inter-flow coded *whenever* possible. Based on the LP output, we compute the ratio of inter-flow versus intra-flow coded traffic, and insert packets into these queues according to these ratios. We also associate a timeout with every packet in Q_{inter}. Once the medium is available for transmission, we poll the first packet from Q_{inter}, denoted as P, and searches for another packet to inter-code with P first from Q_{inter} and then from Q_{intra}. If found, we send out the resulting inter-flow coded packet immediately. Otherwise if P's associated timer has not expired, we instead send out the first packet from Q_{intra}. When the timer expires, we send out P even if it cannot be inter-flow coded with another packet to limit its delay.

7. PERFORMANCE EVALUATION

In this section, we first describe our evaluation methodology, and then present performance results.

7.1 Evaluation Methodology

We implement *O3* and the following protocols in Qualnet 3.9.5 and conduct extensive simulation to compare their performance:

1. Shortest-path routing (SPP) using the ETX routing metric, which minimizes the total number of expected transmissions from a source to its destination [4].
2. Shortest-path routing with rate-limiting (SPP-RL), the same as SPP except the flows' sending rates are optimized using the conflict graph interference model as in [14].
3. COPE, a state-of-art shortest path routing protocol with inter-flow network coding.
4. COPE with rate limiting (COPE-RL), the same as COPE except that the flows' sending rates are optimized using the conflict graph model.
5. MORE, a state-of-art opportunistic routing protocol.
6. Optimized opportunistic routing, also called *O3*-Intra, since it is the same as *O3* except that it disables inter-flow coding.

O3-Intra improves MORE by optimizing opportunistic routing and rate limiting. To our knowledge, this is the first paper that extensively compares single path routing, opportunistic routing, and inter-flow coding with and without rate limiting. The evaluation allows us to not only understand the performance of *O3* but also examine individual benefit of inter-flow coding, opportunistic routing, and rate limiting.

Since SPP uses unicast transmissions, SPP-RL uses a link-based conflict graph model, which represents wireless links as vertices in a conflict graph and draws an edge between two conflict vertices if and only if the corresponding wireless links interfere. Based on this definition, links corresponding to the vertices in a clique of the conflict graph cannot be active simultaneously. Since COPE and all of the opportunistic routing protocols use either pseudo or real broadcast transmissions, we use a node-based conflict graph model, which considers two broadcast transmissions to interfere if either (i) the transmitters carrier sense each other or (ii) anyone of their receivers is interfered by the other transmission.

Both MORE and *O3*-Intra use a batch size of 32 packets, which is the default batch size used in MORE [2]. Further increasing the batch size yields little benefit. *O3* uses a batch size of 16 with 2 outstanding batches to effectively support inter-flow coding.

We use the following network topologies: (i) canonical topologies shown in Figure 3, (ii) 5x5 grid topologies, (iii) 25-node random topologies, (iv) Roofnet topology with 35 nodes [26], (v) UW testbed topologies with 14 nodes [25]. Roofnet is an IEEE 802.11b testbed, whereas UW traces contain measurements from 802.11a and 802.11b testbeds. We also use both 802.11a and 802.11b in the synthetic topologies. Since the results under grid topologies are similar to the other topologies, they are omitted in the interest of brevity.

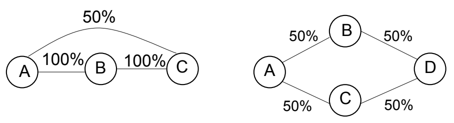
In 802.11a, each sender uses a transmission power of 10 dBm (Qualnet default) and a fixed PHY rate of 6Mbps, which gives 230*m* communication range and 1535*m* carrier sense range. In 802.11b, each sender uses transmission power of 15dBm (Qualnet default) and a fixed PHY rate of 2Mbps, which gives 1027*m* communication range and 3100*m* carrier sense range. We can certainly use another data rate for evaluation and expect similar relative performance. We compute the conflict graph by using these range values to determine if two links or nodes interfere.

Nodes are placed in a 1000*m* x 1000*m* area for 802.11a, and in a 2500*m* x 2500*m* area for 802.11b. In addition, we extend Qualnet to generate directional inherent packet losses. For the testbed topologies, the loss rates are based on the traces. For the synthetic topologies, the loss rates are uniformly distributed either between 0 and 30% (low loss), between 0 and 50% (medium loss), or between 0 and 80% (high loss).

We generate saturated UDP traffic with 1024-byte payload, and vary the number of flows from 1 to 16. Since the choice of routing protocols is important for multihop flows, our simulation randomly picks a source and destination that have at least 2 hops. For single-hop flows, all schemes with rate limiting can simply activate one-hop flows as much as possible and disable interfering multihop flows to achieve maximum throughput and the effects of routing cannot be not reflected. For each scenario, we conduct 10 random runs, each lasting 30 seconds. We report the average total throughput of these runs. In addition, the error bars on the graph show the standard deviation of the sample mean.

	O3	*O3*-Intra	MORE	COPE	SPP-RL	SPP
Linear chain	3.45	2.98	2.78	2.84	2.56	1.78
Diamond	1.50	1.11	0.91	0.47	0.47	0.40

Table 1: Total throughput (Mbps) for the topologies in Figure 3



(a) Linear chain (b) Diamond

Figure 3: Two symmetric flows between the left-most and right-most nodes.

7.2 Performance Results

Canonical Topologies: Table 1 reports the throughput of the two canonical topologies shown in Figure 3. In the linear topology, there are two flows: from A to C and from C to A. Here, we observe that *O3* $>$ *O3*-Intra $>$ COPE $>$ MORE $>$ SPP-RL $>$ SPP. SPP-RL outperforms SPP by 44% due to its proper rate limiting. COPE out-performs SPP by 60% due to inter-flow coding. *O3*, *O3*-Intra, and MORE out-perform SPP by taking advantage of opportunistic routing to effectively combat lossy wireless links. Among them, *O3*-Intra out-performs MORE through optimized rate limiting and opportunistic routing, while *O3* outperforms all the protocols by simultaneously exploiting inter-flow coding, opportunistic routing, and rate limiting. For the diamond topology with two flows, from A to D and from D to A, the relative ranking between various protocols

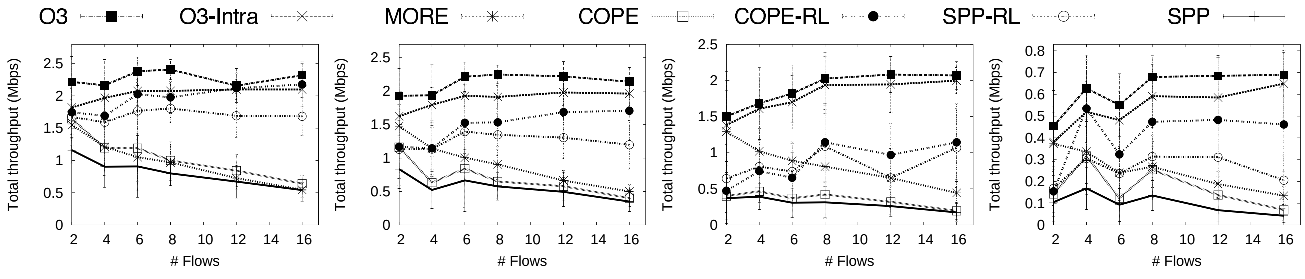

(a) 802.11a, low loss (b) 802.11a, medium loss (c) 802.11a, high loss (d) 802.11b, high loss

Figure 4: Total throughput in 25-node random topologies.

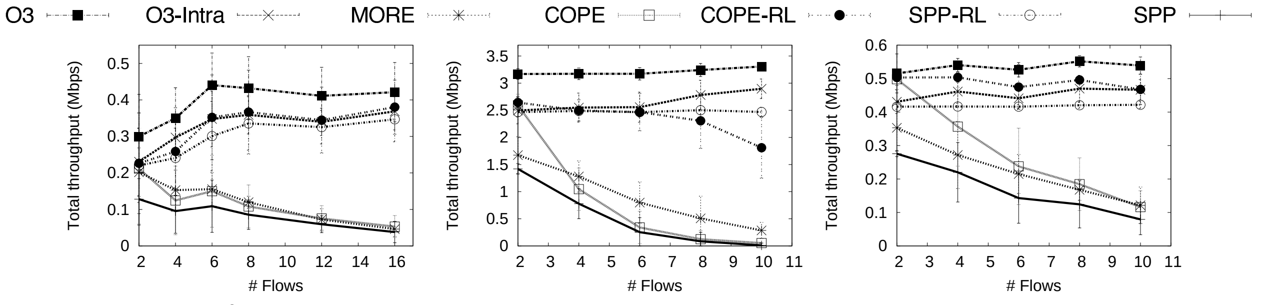

(a) Roofnet 802.11b (b) UW 802.11a (c) UW 802.11b

Figure 5: Total throughput in the testbed topologies.

remains almost the same, except a few differences. Here, COPE performs only slightly better than SPP and similarly to SPP-RL. This is because packet losses on the shortest paths significantly reduce the inter-flow coding opportunities. This also causes MORE to out-perform COPE by 93%. In contrast, *O3* can effectively take advantage of inter-flow coding over lossy wireless links and achieves the best performance. Its benefits over *O3*-Intra, MORE, COPE, SPP-RL and SPP are 35%, 65%, 219%, 219% and 275%, respectively.

Effects of number of flows in synthetic topologies: Figure 4 summarizes the performance results for 802.11a and 802.11b from low to high loss rates. We make the following observations.

First, *O3* out-performs state-of-the-art protocols in all the scenarios. For example, as shown in Figure 4(a), in low loss random topologies, compared with the protocols without rate limits, *O3* has 43-325% gain over MORE, 35-262% gain over COPE, and 92-329% gain over SPP; compared with the protocols with rate limit, *O3* out-performs *O3*-Intra by 3-22%, COPE-RL by 2-29%, and SPP-RL by 32-38%. The performance gain of *O3* comes from opportunistic routing, rate limiting, and inter-flow coding. In particular, we observe (i) *O3*, *O3*-Intra, and MORE out-perform SPP since opportunistic routing can more effectively cope with lossy wireless links, (ii) *O3* and *O3*-Intra out-perform MORE due to their optimized opportunistic routes and rate limits, and (iii) *O3* out-performs *O3*-Intra due to inter-flow coding. Note that the total throughput does not monotonically increase with the number of flows since we randomly select the flow sources and destinations and generate random link loss rates in each run.

Second, rate limiting is important to all the protocols. In all cases, we observe the protocols with rate limiting significantly out-perform their counter-parts without rate limiting. For example, as shown in Figure 4(a), *O3*-Intra out-performs MORE by 18-284%, COPE-RL out-performs COPE by 6-240%, and SPP-RL out-performs SPP by 44-211%.

Third, loss rate has significant impact on the effectiveness of opportunistic routing and inter-flow coding. In particular, as we would expect, the benefits of opportunistic routing increases with link loss rates. For example, comparing the results between low and high loss rates (Figure 5(a) and (c)), we observe that the gap between the performance gain of *O3* and *O3*-Intra over the other protocols increases. Moreover, MORE performs worse than COPE-RL and SPP-RL under low loss rate, and performs better than them under high loss rate because the benefit of opportunistic routing under high loss rate offsets the disadvantage arising from its lack of rate limiting. Moreover, the benefits of inter-flow coding decreases with link loss rates. For example, under high loss rate, *O3* has smaller gain over *O3*-Intra (3-9% gain), COPE performs similarly to SPP, and COPE with rate limiting performs similarly to SPP with rate limiting. Loss rates reduce inter-coding opportunities because when fewer packets are received at each node, they not only limit the choices of inter-flow coding and but also make the next hop harder to decode. Similar effects are observed in 802.11b as shown in Figure 5(d). Nevertheless, *O3* continues to out-perform the other protocols: it out-performs *O3*-Intra by 6-21%, COPE-RL by 17-194%, SPP-RL by 105-235%, MORE by 21-412%, COPE by 99-900%, and SPP by 273-1500%.

Effects of number of flows in testbed topologies: Figure 5(a), (b), and (c) show the performance results under Roofnet with 802.11b 1Mbps, UW testbed with 802.11a 6Mbps, and UW testbed with 802.11b 1Mbps, respectively. In Roofnet, 68% of the links have within 1% loss and 80% of the links have within 57% loss. In UW 802.11a testbed, 75% of the links have within 1% loss and 80% of the links have within 51% loss. In UW 802.11b testbed, 52% of the links have within 1% loss and 80% of the links have within 93% loss. We make the following observations based on the performance results from these testbeds.

First, *O3* $>$ *O3*-Intra, COPE-RL, SPP-RL $>$ MORE, COPE $>$ SPP. The relative orderings of COPE-RL and *O3*-Intra depend on the loss rates: the former performs better under low loss and the latter is better under high loss.

Second, as in the synthetic topologies, all the protocols with rate limiting significantly out-performs their counterparts without rate limiting. For example, in Roofnet *O3*-Intra out-performs MORE by 15-696%, COPE-RL out-performs COPE by 1-617%, SPP-RL out-performs SPP by 71-811%.

Third, *O3* consistently out-performs all the other protocols. As shown in Figure 5(a), in Roofnet, *O3* out-performs *O3*-Intra by 14-30%, COPE-RL by 11-35%, SPP-RL by 21-46%, MORE by 48-810%, COPE by 41-694%, SPP by 134-111%. As shown in Figure 5(b) and (c), in 802.11a and 802.11b UW testbed topologies, *O3* out-performs *O3*-Intra by 14-27%, COPE-RL by 2-83%, SPP-RL by 24-34%, MORE by 46-1000%, COPE by 3-6100%, SPP by 87-32600%.

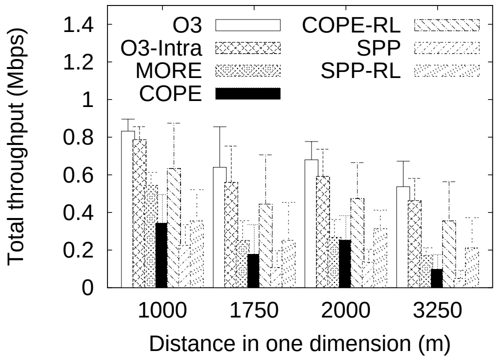


Figure 6: Throughput under varying network density in 25-node 802.11b random topologies (8 flows, high loss).

Effects of network density: Next we vary the network density in 25-node 802.11b random topologies. We vary the area from 1000x1000m^2 to 3250x3250m^2. Figure 6 plots the total throughput. As we can see, *O3* out-performs the other protocols across all network densities. As before, rate limiting leads to significant performance improvement in all the routing protocols.

8. CONCLUSION

Optimizing inter-flow network coding in opportunistic routing is useful but challenging due to the strong interactions between information splitting in opportunistic routing and inter-flow network coding. We approach the problem by proposing a novel overlay framework to decouple opportunistic routing and inter-flow network coding, and develop the first approach to jointly optimize opportunistic routing, rate limiting, and inter-flow network coding. We design a routing protocol to realize its benefit and demonstrate its effectiveness using Qualnet simulation. Furthermore, our simulation reveals the relative benefit of opportunistic routing, inter-flow coding, and rate limiting. Moreover, we hope that our overlay framework is useful and has other interesting wireless applications, which we plan to explore in the future.

Acknowledgments: This research is supported in part by NSF grants CNS-0916106 and CNS-0546755. We thank Alberto Lopez Toledo and anonymous reviewers for their valuable comments.

9. REFERENCES

[1] S. Biswas and R. Morris. ExOR: opportunistic multi-hop routing for wireless networks. In *Proc. of ACM SIGCOMM*, Aug. 2005.

[2] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proc. of SIGCOMM*, 2007.

[3] P. Chaporkar and A. Proutiere. Adaptive network coding and scheduling for maximizing throughput in wireless networks. In *Proc. of ACM MobiCom*, 2007.

[4] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, Sept. 2003.

[5] S. Das, Y. Wu, R. Chandra, and C. Hu. Context based routing: Technique, applications and experience. In *Proc. of NSDI*, Apr. 2008.

[6] H. Feng, Y. Shu, S. Wang, and M. Ma. SVM-based models for predicting WLAN traffic. In *Proc. of IEEE ICC*, 2006.

[7] R. Haemmerle. LP solve – incremental version. `http://pauillac.inria.fr/~haemmerl/lp_solve_inc/`.

[8] 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*, Sept. 2006.

[9] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control in communication networks: Shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 1998.

[10] T. Kim, S. Vural, and I. Broustis. A framework for joint network coding and transmission rate control in wireless networks. In *Proc. of IEEE INFOCOM*, 2010.

[11] D. Koutsonikolas, Y. C. Hu, and C. Wang. XCOR: Synergistic interflow network coding and opportunistic routing. In *MobiCom SRC*, 2008.

[12] D. Koutsonikolas, C.-C. Wang, and Y. C. Hu. CCACK: efficient network coding based opportunistic routing through cumulative coded acknowledgments. In *Proc. of IEEE INFOCOM*, 2010.

[13] J. Le, J. Lui, and D. Chiu. DCAR: distributed coding-aware routing in wireless networks. *IEEE Transactions on Mobile Computing*, 2010.

[14] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, and E. Rozner. Predictable performance optimization for wireless networks. In *Proc. of ACM SIGCOMM*, Aug. 2008.

[15] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, Z. Zhong, G. Deshpande, and E. Rozner. Effects of interference on throughput of wireless mesh networks: Pathologies and a preliminary solution. In *Proc. of HotNets-VI*, Nov. 2007.

[16] Y. Lin, B. Li, and B. Liang. CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding. In *Proc. of ICNP*, Oct. 2008.

[17] M.-H. Lu, P. Steenkiste, and T. Chen. Design, implementation, and evaluation of an efficient opportunistic retransmission protocol. In *Proc. of ACM MobiCom*, 2009.

[18] D. Lun, M. Medard, and R. Koetter. Network coding for efficient wireless unicast. *2006 International Zurich Seminar on Communications*, 2006.

[19] A. K. Miu, H. Balakrishnan, and C. E. Koksal. Improving loss resilience with multi-radio diversity in wireless networks. In *Proc. of ACM MobiCom*, 2005.

[20] A. K. Miu, G. Tan, H. Balakrishnan, and J. Apostolopoulos. Divert: Fine-grained path selection for wireless LANs. In *Proc. of ACM MobiSys*, 2004.

[21] J. Moy. OSPF version 2. In *Internet Engineering Task Force, RFC 2328*, Apr. 1998. http://tools.ietf.org/html/rfc2328.

[22] C. Qin, Y. Xian, C. Gray, N. Santhapuri, and S. Nelakuditi. I2MIX: Integration of intra-flow and inter-flow wireless network coding. In *Proc. of WiNC*, 2008.

[23] 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*, Apr. 2008.

[24] B. Randunovi and J. Y. L. Boudec. Rate performance objectives of multihop wireless networks. In *Proc. of IEEE INFOCOM*, Apr. 2004.

[25] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. of ACM SIGCOMM*, 2006.

[26] MIT Roofnet. http://www.pdos.lcs.mit.edu/roofnet/.

[27] B. Scheuermann, W. Hu, and J. Crowcroft. Near-optimal coordinated coding in wireless multihop networks. In *Proc. of CoNEXT*, 2007.

[28] S. Sengupta, S. Rayanchu, and S. Banerjee. An analysis of wireless network coding for unicast sessions: The case for coding-aware routing. In *Proc. of IEEE INFOCOM*, Apr. 2007.

[29] F. Soldo, A. Markopoulou, and A. Toledo. A simple optimization model for wireless opportunistic routing with intra-session network coding. In *Proc. of IEEE NetCod*, Jun. 2010.

[30] J. J. T. Ho and H. Viswanathan. On network coding and routing in dynamic wireless multicast networks. In *Proc. of Workshop on Information Theory and its Applications*, 2006.

[31] Y. Yan, B. Zhang, H. T. Mouftah, and J. Ma. Practical coding-aware mechanism for opportunistic routing in wireless mesh networks. In *Proc. of ICC*, 2008.

[32] Y. Yuan, H. Yuan, S. H. Wong, S. Lu, and W. Arbaugh. ROMER: resilient opportunistic mesh routing for wireless mesh networks. In *Proc. of IEEE WiMESH*, Sept. 2005.

[33] S. Yun and H. Kim. Rate diverse network coding: Breaking the broadcast bottleneck. In *Proc. of ACM MobiHoc*, 2010.

[34] J. Zhang, Y. P. Chen, and I. Marsic. Network coding via opportunistic forwarding in wireless mesh networks. In *Proc. of IEEE WCNC*, 2008.

[35] X. Zhang and B. Li. Optimized multipath network coding in lossy wireless networks. In *Proc. of ICDCS*, 2008.

[36] Z. Zhong and S. Nelakuditi. On the efficacy of opportunistic routing. In *Proc. of IEEE SECON*, Jun. 2007.