Computer Networks 54 (2010) 196–207

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

MAC-layer proactive mixing for network coding in multi-hop wireless networks

Jian Zhang [a,*], Yuanzhu Peter Chen [b], Ivan Marsic [a]

[a] Department of Electrical and Computer Engineering, Rutgers University, 96 Frelinghuysen Road, Piscataway, NJ 08854-8088, United States
[b] Department of Computer Science, Memorial University of Newfoundland, Canada

ARTICLE INFO

Article history:
Available online 30 June 2009

Keywords:
Network coding
802.11 MAC
Multi-hop wireless network

ABSTRACT

Network coding is a recent research topic in wireless networking. By combining multiple packets in a single broadcast transmission, network coding can greatly improve the capacity of multi-hop wireless networks. Packet mixing, when applied with traditional routing, can only be performed at the junctions of the paths determined by the routing module. This limits significantly the coding opportunities in the network. This paper presents a novel MAC-layer mixing method, named BEND, which proactively seizes opportunities for coding. Without relying on fixed forwarders, BEND allows each node in the neighborhood to be a potential coder and forwarder and coordinates their packet transmissions for higher coding gain. By taking advantage of redundant copies of a packet in the neighborhood coding repository, the number of mixing points, and thus the coding opportunities, can be significantly increased. This high coding gain is verified by our simulation studies.

1. Introduction

Network coding is a relatively new research area in communication networks. It enables data flows to approach the Shannon Capacity Limit individually by splitting and combining information at intermediate nodes in the network [6]. Such operations on information flows can be implemented as simple linear combinations over some finite field. Two fundamental benefits of network coding are greater throughput and higher robustness. These in turn translate to energy efficiency and fault tolerance in multi-hop wireless networks. Current research on network coding is transitioning from theoretical frameworks to increasingly practical systems.

The way that network coding increases the throughput of a multi-hop wireless network can be explained using a simple example of a 5-node network (Fig. 1) [10]. Here, nodes X, B, and O are within each other's transmission range; so are nodes Y, A, and O. Suppose that node X has a packet p_1 for Y via O and node A has a packet p_2 for B via O. In a traditional non-coding approach (Fig. 1a), after O's reception of packets p_1 and p_2, it relays these packets separately. Thus, a total of 4 transmissions are required. In contrast, if network coding is used (Fig. 1b), after O's reception of p_1 and p_2, it transmits XOR combination $p_1 \oplus p_2$ in the wireless channel. Since node B (Y, respectively) is within the transmission range of X (A, respectively), it has also overheard p_1 (p_2, respectively). With node B's knowledge of p_1 (Y's knowledge of p_2, respectively), it can reconstruct p_2 (p_1, respectively) by applying XOR $\oplus$ to the two receptions from X (A, respectively) and O. Consequently, only 3 transmissions are needed for the packet exchange. More generally, network coding can be used in such scenarios as a path transporting two flows in reverse directions (Fig. 2a) and combining multiple packets (Fig. 2b).

In a previous wireless coding approach, COPE [10], packet mixing can only be performed at the joint nodes of the paths determined by the routing module, such as the focal nodes in Figs. 1 and 2. This significantly limits the coding opportunities in the network. Clearly, in order for network

* Corresponding author. Tel.: +1 732 762 1641.
E-mail address: jianzhangru@gmail.com (J. Zhang).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.04.013

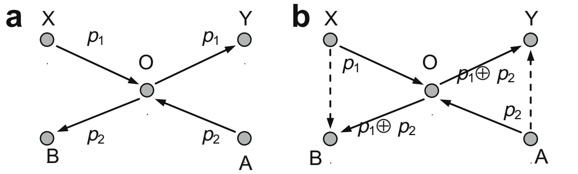


Fig. 1. Wireless network coding illustrated: (a) regular exchange; and (b) coded exchange.

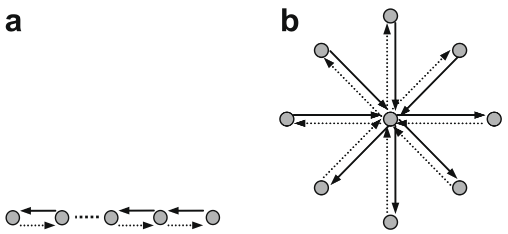


Fig. 2. More general coding scenarios: (a) chain; and (b) wheel.

coding to be useful in multi-hop wireless networks, there should exist sufficient opportunities to mix traffic flows in the network. Currently, this is achieved by concentrating flows at certain nodes in the network. This could be implemented via centralized coding-aware routing [12] or using coding-aware metrics [14]. As a result, some nodes in the network are favored by the routing module so that much traffic is routed through them for more coding opportunities. However, these approaches can be problematic. First, a network layer implementation of such a "traffic-sensitive" routing protocol is unrealistic in multi-hop wireless networks since traffic flows change over time. Routing that depends on the correlation of dynamic flows has been shown impractical, even in the Internet where the traffic is much more statistically stable over time [1,11]. Further, concentrated traffic inevitably overloads intermediate nodes in the network. These overloaded forwarders can be a vulnerable point because of higher risk of battery-energy depletion and information leaks. Other problems of traffic concentration include increased queuing delay and thus end-to-end delay, danger of buffer overflow, and further adversary effects to TCP flows. At the link layer, on one hand, traffic concentrated within a neighborhood worsens channel contention in the area. On the other hand, if flows are forced to go through a specific node, this overloaded node is bound to drop packets which it is unable to handle. This is especially problematic in multi-hop wireless networks because dropping packets along a path means invalidating the work performed by earlier forwarders and wasting the network bandwidth already consumed. The benefit of being able to scatter flows through multiple forwarders dynamically at the link layer in a multi-hop wireless network is called "*diffusion gain*" in the rest of the paper.

Indeed, traffic separation rather than concentration has been a key approach to higher throughput in mesh networks. When flows are more evenly distributed in the network, the interference among them is minimized and the network capacity limit can be approached [7]. Hence, traffic concentration in network coding conflicts the need of traffic separation. Does traffic mixing for network coding inevitably imply traffic concentration? Not really.

In this article, we present BEND, a MAC layer solution to practical network coding in multi-hop wireless networks. It is the first exploration of the broadcasting nature of wireless channels to proactively capture more coding opportunities. As a matter of fact, the result of a node's transmitting a packet is that all of its neighbors can potentially receive it, and such redundancy of packets should and can be utilized. In BEND, any node in the network can code and forward a packet even when this node is not the intended MAC receiver of the packet, if the node believes that doing so it can lead the packet to its ultimate destination. Essentially, BEND considers the union of the contents of the interface queues of the nodes within a neighborhood collectively, i.e. a "neighborhood coding repository", whereas traditional mixing methods, e.g. COPE, only process "individual coding repositories" at separate nodes. Our experimental evaluation shows that BEND creates significantly more coding opportunities in a dynamic and adaptive fashion with minimum assumptions on the routing protocol compared to prior work. The contributions of BEND are:

(1) It makes network coding practical by proactively seizing such opportunities and by using them intelligently.
(2) This is achieved without concentrating traffic flows or overloading specific nodes. It exploits the broadcasting nature of wireless channels by utilizing redundant packet copies within the proximity of a node. In this way, it achieves both diffusion gain and coding gain, which are conflicting in the existing solutions.
(3) It exploits another dimension of multi-user diversity in wireless networks. Multi-user diversity has proved to be effective in achieving higher aggregate system performance in wireless communications. Here in BEND, multi-user diversity is in the sense of diversity of queue contents at different forwarders.

The rest of the paper is organized as follows. In Section 2, we review the basic idea of BEND to help readers with the subsequent relatively involved details. We then highlight the design objectives of BEND and the challenges in Section 3. The design details are presented in Section 4. The effectiveness of BEND is tested by the experiments in Section 5. After digesting the details, the readers are walked through a discussion in the context of some recent related work on practical network coding and on exploration of the broadcasting nature in multi-hop wireless networks in Section 6. We conclude this paper in Section 7 and speculate on future research to further explore BEND.

2. Basic idea

The gist of BEND is to utilize overheard packets that are otherwise discarded in conventional networking protocols.

In a network supporting multiple flows, there are various loci where two flows come close. Fig. 3 depicts an example of such a local area of the network. In the figure, packet p_1 goes from node X via A to Y, and another packet p_2 goes from node U via C to V. These routes are determined by the routing protocol. In a multi-hop wireless network, some other nodes, say B_1, B_2, and B_3 can overhear the transmissions of p_1 and p_2. Traditional methods simply discard these overheard packets to avoid duplication, thus missing potential coding opportunities. Instead, BEND seizes the coding opportunities by enabling any one of B_1, B_2, or B_3 to forward $p_1 \oplus p_2$. This is a novel idea for exploiting the broadcasting nature of wireless channels. Once a packet, such as p_1 (p_2, respectively) in this example, is transmitted, it is in effect received by all neighbors of the transmitter, i.e. nodes A, B_1, B_2 and B_3 (nodes B_1, B_2, B_3 and C, respectively). Instead of wastefully discarding the overheard packets, BEND stores them at the MAC layer and uses them later. In this way, these nodes share a significantly richer repository for coding by collectively snooping data communications in the neighborhood. BEND coordinates the coding and forwarding of the queued packets so that these nodes make use of such a repository jointly. The benefit is to enable many more coding opportunities in the neighborhood, without forcing traffic flows through a fixed joint node, as required by COPE.

To use an analogy, a packet experiences such a proactive mixing of packets similar as the light photons experience the bending of a gravitational field. Here, the “gravity” for a packet arises from the possibility of combining it with other packets on potential forwarders en route. At each moment, the queues of the forwarders are likely to contain different packets due to the spatial diversity (e.g., forwarder positions), and the temporal diversity (e.g., traffic dynamics). Since a packet is likely to be stored at several forwarders, it tends to be forwarded by the one at which higher-gain coding (coded up with more packets) or any coding can be achieved. Due to the dynamics of the packets overheard and stored by nodes at different positions in the network, such tendency can change on a per-packet and per-hop basis. Therefore, a MAC-layer per-packet adaptation shows great applicability and flexibility in seizing coding opportunities. Globally, a packet’s trajectory follows *only approximately* the exact route specified by the routing protocol. Each time a packet moves forward, it is transmitted by a node “around” the route determined by the routing module. For example in Fig. 4, there is a flow between nodes S and D and its route is determined by the routing module as indicated by the thick light-colored line. Consider three back-to-back packets, p_1, p_2 and p_3. They can take different trajectories as shown in the figure to be combined with packets from other flows in the network in different times and at different mixing nodes. This flexible and real-time adaptation offers high coding ratio and, hence, throughput gain. We believe such per-packet and per-hop decision-making is only feasible through a MAC layer implementation.

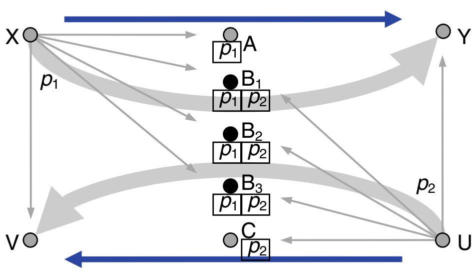


Fig. 3. Neighborhood packet repository.

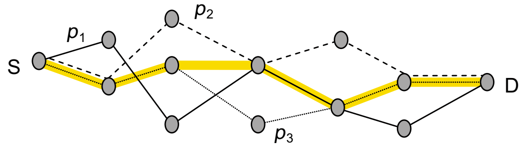


Fig. 4. Closure of trajectories.

3. BEND objectives and challenges

We design BEND with the following objectives:

- BEND is based on IEEE 802.11 MAC [8] and should be easy to implement for practical use. Therefore, it should follow the 802.11 CSMA/CA paradigm and achieve reliable delivery for each transmission.
- As a link-layer solution, BEND should work with a routing protocol instead of making routing decisions. Indeed, choosing what type of routing protocol, proactive or not, source routing or distance vector, flat or hierarchical, position-based or energy-aware, is not necessarily only a performance issue, and should be left to the network operator. To that end, BEND should make minimum assumptions about the routing protocol used.

In order to design such an efficient mixing protocol, we must address the following challenges:

3.1. Maximizing coding chances

To promote coded transmissions for throughput gain, a mechanism is needed to ensure that packets have a better chance to be coded and transmitted by one forwarder than to be transmitted non-coded. This must be handled without starving any flows or nodes in the network.

3.2. Recognizing coding conditions

When a node has a packet to forward, it needs to know whether coding it with other queued packet(s) may save bandwidth; i.e. it needs to determine if the receivers can decode the coded packets.

3.3. Duplication of packets

All nodes operate in the promiscuous mode for opportunistic forwarding. As a result, a packet will be overheard and queued at multiple neighbors. There must be a mechanism to ensure that it is forwarded by only one of these neighbors.

3.4. Reliable link-layer broadcast

Since a coded packet is intended for multiple receivers, an efficient and reliable link-layer broadcasting mechanism is needed as a building block.

These challenges are addressed in the next section where details of BEND are presented.

4. Design details

In this section, we present the main components in the design of BEND. The basic operation of BEND is illustrated by a simple example in Fig. 5 although BEND works under much more general conditions. In Fig. 5a, node X has packet p_1 for node Y that is two hops away, and node U has packet p_2 for node V, also two hops away. The forwarders determined by the routing protocol are nodes A and C, respectively. We further assume that three other nodes, B_1, B_2, and B_3, are also within the range of nodes X, Y, U, and V. When a packet, say p_1 or p_2, is handed from the network layer down to the MAC layer, its header is enhanced (Section 4.1 below) to include not only the address of the next-hop node but also that of the following-hop node. Such information can be obtained by querying the routing module (Section 4.2). After node X's packet p_1 and node U's packet p_2 are transmitted, p_1 is received by nodes A (intended forwarder), B_1, B_2, B_3 and V, and p_2 is received by nodes B_1, B_2, B_3, C (intended forwarder), and Y. For p_1, it is placed in the queues of nodes A, B_1, B_2, and B_3 because they are all neighbors of p_1's second-next-hop (node Y) as indicated by the packet header. Otherwise, it is buffered by node V for future decoding. Similarly, p_2 is queued at nodes B_1, B_2, B_3 and C, and buffered at node Y. Nodes B_1, B_2 and B_3 can choose to transmit $p_1 \oplus p_2$ if they determine that the coded packets can be correctly decoded by their second-next-hop neighbors (Section 4.3). All of the intermediate nodes A, B_1, B_2 and B_3 and C could forward the packet(s) in their queues, coded or not. In order to expedite the packet forwarding, coded packets are transmitted with a higher priority, without starving uncoded packets (Section 4.4). Assume that node B_2 wins the channel and transmits $p_1 \oplus p_2$ (Fig. 5b). The second-next-hop nodes V and Y receive the XORed packets and are able to decode them using the packets stored in their buffer. Then they immediately reply with an ACK in a "distributed bursty" fashion in the order specified by the enhanced MAC header. Such a reliable link-layer broadcast mechanism (Section 4.5) also helps to remove the packets queued at the intermediate nodes to avoid packet duplication (Fig. 5c).

4.1. Header specification

BEND performs packet coding and tagging at a MAC sender. It requires a modification to the DATA and ACK headers of the existing 802.11 MAC Specifications [8].

In Fig. 6, we show the header fields modified or added for BEND. The header of DATA frame may have a different format depending on whether the payload is encoded. If non-coded, in addition to the sender address (*SA*) and receiver address (*RA*), the header includes the IP address of 2nd-next-hop (described in Section 4.2 below). If encoded, it has a list *RA*[] of receiver addresses, and corresponding list *packetID*[] for all the encoded packets. Each *RA* is 6-byte long. The packet ID is generated by creating a 4-byte hash value out of the source's IP address and the sequence number carried by the IP packet, as in COPE [10]. A 2-bit type and a 4-bit sub_type field in the frame-control field specify frame types, i.e., non-coded DATA, encoded DATA, ACK, NACK or other 802.11 frame.

Each ACK or NACK contains an *SA* and the packet ID of the original packet to acknowledge. Notice that BEND uses *SA* in ACK instead of *RA* as in the 802.11 Specifications. The reason for this is described in Section 4.5 below.

4.2. 2nd-next-hop en route

When a node requests help from its neighbors to forward a packet, it finds the IP address of the 2nd-next-hop (denoted by 2NH in the rest of the paper) along the path to the destination. If the destination is at least two hops away, it sets the 2nd-next-hop field in MAC header and transmits this DATA frame. This tells the potential forwarders where this packet should go next. The knowledge of 2NH is provided by the routing module. If a source or link-state routing is used, this is trivial. However, such knowledge is not immediately available for distance-vector based routing protocols. 2NH information can be obtained by minor modifications to distance-vector

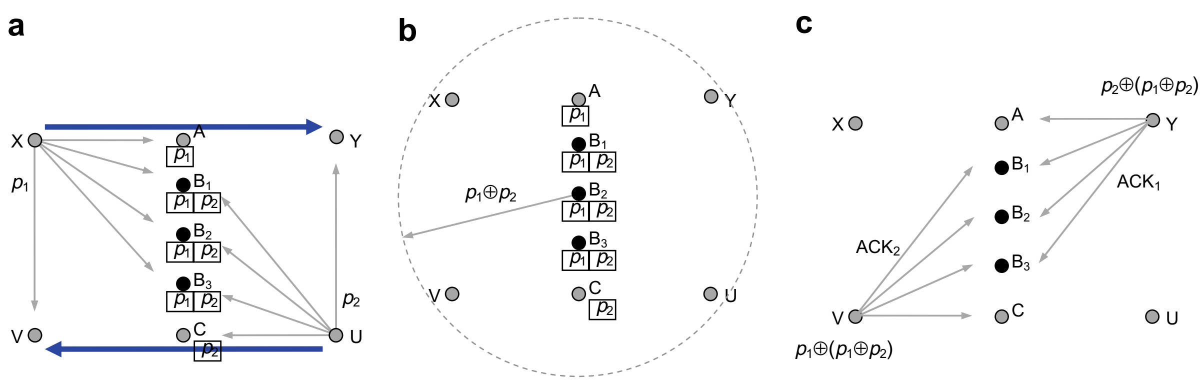


Fig. 5. BEND – design overview.

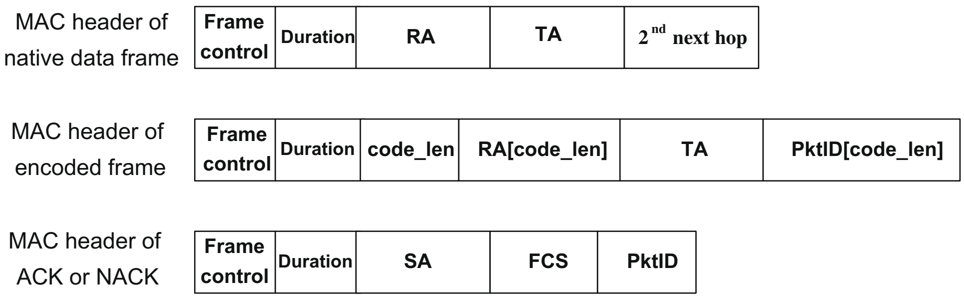


Fig. 6. MAC headers of BEND.

protocols. We simply add a "via" field to each distance vector in routing packets. That is, in the routing table broadcast to the neighbors, each entry destination is associated with distance estimation plus the neighbor via which this distance is established.

Upon receiving a DATA frame, only the nodes that are neighbors of the 2NH specified in the frame header are allowed to forward it. This guarantees that the packet propagation is restricted within a "stripe" along the route without flooding the network. When such a packet p_1 is received by a potential forwarder, it is passed up to the network layer with the 2NH information. The network layer fills in the next-hop field using the specified 2NH information and sends it down to the queue. BEND then searches the queue for mixing opportunities with other queued packets.

4.3. Queuing and mixing strategy

At each intermediate node, multiple packets can be mixed in a single transmission. For each pair of packets p_1 and p_2 in the set of packets to be combined, they must satisfy the following criteria:

(1) The next-hop receiver of p_1 is p_2's previous forwarder, or one of its neighbors.
(2) The next-hop receiver of p_2 is p_1's previous forwarder, or one of its neighbors.

The first condition ensures that the receiver of p_1 has p_2 in its buffer so that it can XOR p_2 with the coded packet. Likewise, the second condition ensures p_2 can be obtained by its corresponding receiver. These conditions are based on the assumption that a node's neighbors can receive its packets with a reasonable success ratio. Such link delivery ratio can be obtained by a network-layer routing metric, such as ETX [5]. For example, in Fig. 5a, p_1 and p_2 are queued at B_1. Here, p_1's next-hop receiver is *Y*, which happens to be a neighbor of p_2's previous forwarder *U*. Similarly, p_2's next hop is *V*, which is a neighbor of p_1's previous forwarder *X*. Thus, when p_1 and p_2 are encoded, to reconstruct p_1 node *Y* can XOR the coded packet with p_2, which was overheard from *U* earlier. Node *V* performs a similar operation to extract p_2. The probability that both *Y* and *V* can successfully decode the coded packet is a product of the delivery ratios of link *XV* and link *UY*. Therefore, as long as such decoding probability is higher than some threshold, we consider that the mixing criteria are met.

To realize the above conditions, we need to maintain at each node a 1-hop-neighbor and neighbor's-neighbor lists. These lists along with the link delivery information can be easily constructed and updated based on the routing packet exchanges with the above "via" extension.

Each node stores packets that are intended for itself (with matching *RA*) and the packets that are overheard in different FIFO queues, denoted by Q_1 and Q_2, respectively. Those that satisfy the coding conditions are moved to a queue, denoted as *mixing-Q*. The packet matching process is as follows.

When a packet p_1 is passed down from the network layer, BEND searches the mixing-Q, Q_1 and Q_2 in this order for coding partners. BEND tries to mix as many packets as possible into a single coded transmission. The more packets are coded in one transmission, the higher the throughput gain achieved. Thus, it always starts the search with the mixing-Q. The condition of mixing more than two packets is that any two packets should satisfy the above pair-wise matching conditions. For example, suppose there are already two packets, p_2 and its coding partner p_3, in the mixing-Q. If pairs (p_1, p_2) and (p_1, p_3) further satisfy the matching criteria, we store p_1 in the mixing-Q along with p_2 and p_3. They are grouped by a linked list. Otherwise, if no other partnerships, i.e. groups of two or more codable packets, can be found in the mixing-Q, BEND searches Q_1 and Q_2 in turn for 2-packet coding opportunities. It starts from the head of the queue, and the first matching packet will be removed and queued along with p_1 at the tail of the mixing-Q. If no matching can be found for p_1, it will be queued at the tail of Q_1 if this node is the intended forwarder, or Q_2 otherwise. The packets will be kept in queue for subsequent matching attempts until they are finally transmitted. If a packet is still alone when scheduled, it will be transmitted non-coded. Again, the forwarder will set its 2NH field so that other nodes down the path can code it.

It might be a concern that above matching mechanism could be slow and costly. COPE assigns a virtual queue for each next hop receiver. This speeds up the matching process since it only needs to search packets at the head of the queues. However, as shown by the mixing criteria, whether packets can be coded together depends not only on their next-hop receivers but on their previous forwarders. So for an implementation considering both matching cost and coding ratio, it needs to maintain a queue for each combination of previous forwarder and next-hop receiver. Although such virtual queuing scheme can accelerate the mixing strategy execution, it imposes much larger demand for memory space and more complicated queue management, especially if this is to be implemented based on a single consecutive address space of a piece of hardware. With our implementation, the computation time is linear to the length of the queue, but it is space efficient. These two approaches are a trade-off between time and space budget and can be balanced under different hardware conditions.

4.4. Two-level prioritization

A packet and its copies could be queued up at different nodes (either the intended forwarder or potential forwarder helpers). The diversity among the forwarder nodes provides the packet various options to be combined with different numbers of and sets of packets, or not coded at all. To maximize coding opportunities, BEND gives coded transmissions higher priority in scheduling. This is achieved at two levels: within a node and among a set of contending nodes.

In a loaded network, end-to-end delay is dominated by queuing delays at individual nodes. Since the coding opportunity is transient, BEND is designed to seize these opportunities effectively. In a forwarding node, the mixing-Q is assigned a higher weight or priority than Q_1 and Q_2. The scheduler generates a random number uniformly between 0 and 1. If the number is greater than W_X and the mixing-Q is not empty, the node retrieves the head packet, dequeues the other packet in the group and combines them in an encoded transmission. Otherwise, it schedules a non-coded packet. With these tunable weights W_X, BEND gives encoded packets better chances for transmission and yet does not starve the non-coded packets without coding opportunities.

When the forwarder nodes contend with each other to transmit their scheduled packets, BEND prioritizes them by assigning them different back-off durations before medium access based on the types of their packets. It is implemented through an EDCF-like type-specific mechanism of IEEE 802.11e [9]. The 802.11e EDCF regulates that, after a node decides to send a type of packet, it must back off for a fixed period (AIFS) and another time interval uniformly distributed between 0 and *cw*, where *cw* is a changing contention window size, to coordinate contending nodes. Initially, *cw* is set to CW_{Min} and is doubled every time a transmission attempt fails until it reaches the specified CW_{Max}. The smaller CW_{Min}, CW_{Max} and AIFS are, the higher priority given to the packet. As in Table 1, we assign a higher access priority to transmissions that could achieve higher coding gain. The prioritization is important in BEND to achieve high throughput for two reasons. First, it coordinates potential forwarders' accesses to make best use of the coding repository. The nodes with more efficient combination can capture the media with higher possibilities. Second, the proactive mixing and forwarding in BEND may incur more medium access attempts in the area and thus more intense contention. Transmission classification and prioritization are necessary to effectively alleviate such contention and reduce the number of collisions.

Table 1
Parameters for packet prioritization.

Type	CW_{Min}	CW_{Max}	AIFS
Overheard non-coded	99	2047	7 × slot time + SIFS
Intended non-coded	63	1023	4 × slot time + SIFS
2-Packet coded	41	1023	3 × slot time + SIFS
3-Packet coded	23	63	2 × slot time + SIFS
x-Packet coded ($x > 3$)	9	63	2 × slot time + SIFS

We use the specific priority settings as in Table 1 and set W_X to 0.2 in all our experiments. However, the priority settings can be finer-tuned and determined by other important factors, such as delay requirement of traffic, or size and content of the queues. For example, if the historical statistics suggest no coding opportunities, the medium access delay for the non-coded packets can be lowered for better performance. The optimization of these parameters and its impact in different scenarios is left to future research.

When packets for mixing are scheduled, they are coded by XOR and the result is encapsulated with a MAC header for encoded frames (Fig. 6). The number of packets encoded is specified in the *code_len* field. Their packet IDs and corresponding receiver addresses are also attached in the fields *PktId*[] and *RA*[] in the header. Then, the forwarder transmits this coded packet and waits for replies from the receivers.

4.5. Decoding, acknowledgement and retransmission

When a coded packet arrives at a receiver, the receiver checks whether its MAC address is in the *RA*[] list in the header. If so, it uses the positions of the other receivers to get the IDs of their packets from the *PktId*[] list. These packets are retrieved from this receiver's buffer and used to extract the packet intended for this receiver. These stored packets had either been forwarded or originated by this node before, or they had been overheard by this node when transmitted as non-coded over the medium. Again, the mixing strategy of Section 4.3 ensures that this node was in the neighborhood of the transmitters so it can hear them with reasonable probabilities. All the packets for decoding are stored in an FIFO buffer. If all packets for decoding are found, the node then decodes its non-coded packet and proceeds to send an ACK. Otherwise, it returns a NACK.

Since a coded packet is broadcast to multiple receivers, the link layer is responsible for the reliability of the broadcast. The 802.11 Specification only includes an unacknowledged, and thus unreliable, broadcast mechanism. Prior work, e.g. COPE [10], resorts to an approximate reliability. Here, we devise a reliable link-layer broadcast. In essence, all receivers of a coded packet are polled by the sender in the order specified in the *RA*[] field of the coded DATA header. So, the receivers send their ACKs back-to-back to the sender without collision.

In addition to reliable link transmission, another important task of ACK is to avoid packet duplication. In BEND, an

ACK is used to free all copies of the delivered packet at the previous forwarders. To do that, the ACK contains the MAC address of the ACK sender instead of the receiver as in the regular 802.11 and the ID of the received non-coded data packet. When a node receives the ACK, it searches for the corresponding packet in the queue using packet ID in the ACK frame. If the ACK's sender (*SA*) is the next-hop node of the data packet, which means that the packet has already been successfully received by its next-hop receiver, this packet can be removed to avoid duplication.

The forwarder of the coded transmission will retransmit the NACKed or non-responded (timeout) packets. For the NACKed, it has to be retransmitted non-coded since it cannot be decoded with current combination. If there are no replies at all from any receivers, it is very likely that there was a collision. Thus, it increases its back-off time based on Table 1 and transmits the same coded packet again.

Once a packet is successfully decoded by an intermediate node, the node can either mix it with other packets in the queue, if there are any coding opportunities, or forward it as a plain packet to next-hop potential forwarders/mixers. This cycle repeats until the packet is delivered to the destination. Therefore, a packet could possibly be coded several times on different intermediate nodes with other packets from various flows. With such implementation, BEND promotes network coding among the traffic in the network.

5. Performance evaluation

We use ns-2, a packet-level simulator, as a basis to test BEND's performance in various scenarios and compare it with IEEE 802.11 and COPE-Sim (an ns-2 implementation of COPE), to find how effective they are in supporting multiple flows in multi-hop wireless networks. We measure the aggregate throughput gain of BEND over COPE-Sim and over 802.11, and investigate how this gain is achieved through other measures and what can affect such a gain.

Our PHY layer model adopts BER (bit error rate) to introduce random packet loss to simulate more realistic operation conditions. Here, we use a BER of 2×10^{-6} so that, after an interface has received a packet, even if its strength exceeded the reception threshold, it may still be dropped with a probability. We fix the data rate at 1 Mbps, the basic rate, without any rate adaptation, although any other fixed data rate would not change the relative performance among the protocols under test. With the two-ray propagation model in ns-2, the transmission range in this case is 250 m. The data flows in the network are all CBR flows of 1000-byte datagrams and with an arrival interval of 0.01 s and duration of 100 s. In general, in each tested scenario, the combination of the flows saturates the network to test the protocols' maximum transportation capabilities. The network uses DSDV to determine routes between sources and destinations.

We test a set of scenarios with different characteristics to investigate BEND relative to COPE-Sim and 802.11. We start with a 3-tier scenario to test the coding capability with multiple flow pairs. Next, we use a cross topology to observe how BEND and COPE can seize the chances of coding 3 or 4 packets in a single transmission. Then we generalize to a 5×5 mesh topology with randomly deployed flows to investigate the effect of hop length and number of flows on these three protocols.

5.1. 3-tier topology

In a 3-tier network, tiers 1 and 3 each consist of 4 nodes, and tier 2 may contain 1, 2, 3, or 4 nodes, referred to as 4–1–4, 4–2–4, 4–3–4, 4–4–4 topologies, respectively. We set the separation distance between tiers to 200 m so that flows between tiers 1 and 3 must use tier-2 node(s) as forwarders. The distance among nodes of the same tier is small. In each of the four topology variants, we place 4 CBR flows randomly between tiers 1 and 3, two in the forward (left-to-right) direction and two in the reverse (right-to-left) direction. Since there are more routes between a source and a destination when we increase the number of tier-2 nodes, the chance that a forward flow and a reverse flow cross at a common forwarder decreases. On the other hand, when more forwarders are available in tier-2, this stage will become less of a bottleneck because, if the flows are routed via different nodes of tier-2, these forwarders collectively will have a better chance to capture the wireless channel than the case when all flows must be routed through a single forwarder as in 4–1–4. Hence, a higher diffusion gain is achieved.

We first measure the aggregate throughput that sums the number of packets arrived at the four UDP receiving agents. The plot in Fig. 7a compares the throughputs of BEND, COPE-Sim and 802.11 in the four topology variants. As seen, 802.11 achieves a higher throughput when increasing the number of tier-2 nodes thanks to the higher diffusion gain due to more forwarders. For COPE-Sim, when it enjoys higher diffusion gain introduced by additional tier-2 nodes, it loses its coding power due to load scattering. BEND, however immediately utilizes the maximum benefit since adding the second forwarder. The throughput gains of BEND and COPE-Sim over 802.11 are plotted in Fig. 7b. Here, we define the throughput gain of a protocol over 802.11 as the ratio of the throughput of the protocol to that of 802.11 minus 1. For the 4–1–4 topology, where all flows go through the single tier-2 node, both COPE-Sim and BEND can almost double the network throughput by applying network coding at this forwarder. In contrast, when there are at least 2 nodes in tier-2 to provide alternative paths, BEND (55–97%) offers nearly double throughput gain over 802.11 compared to COPE-Sim (29–51%). This consistently higher-gain of BEND is realized by allowing tier-2 nodes to transmit more coded packets even if the flows do not necessarily cross at a single node as in the 4–1–4 configuration. To verify this, we record the coding ratio, the number of packets forwarded as coded to the total number of packets forwarded by the tier-2 nodes, for the four topology variants (7c). For COPE-Sim, the coding ratio is lost by about two thirds (from 94% to 38%) as the focal nodes vanish among the forwarders; but BEND manages to lose just slightly over one third (from 94% to 57%). Note that bars in 7 indicate that, among the repeated simulation runs, BEND has much smaller variances than COPE-Sim and 802.11 in throughput and coding ratio. That is,

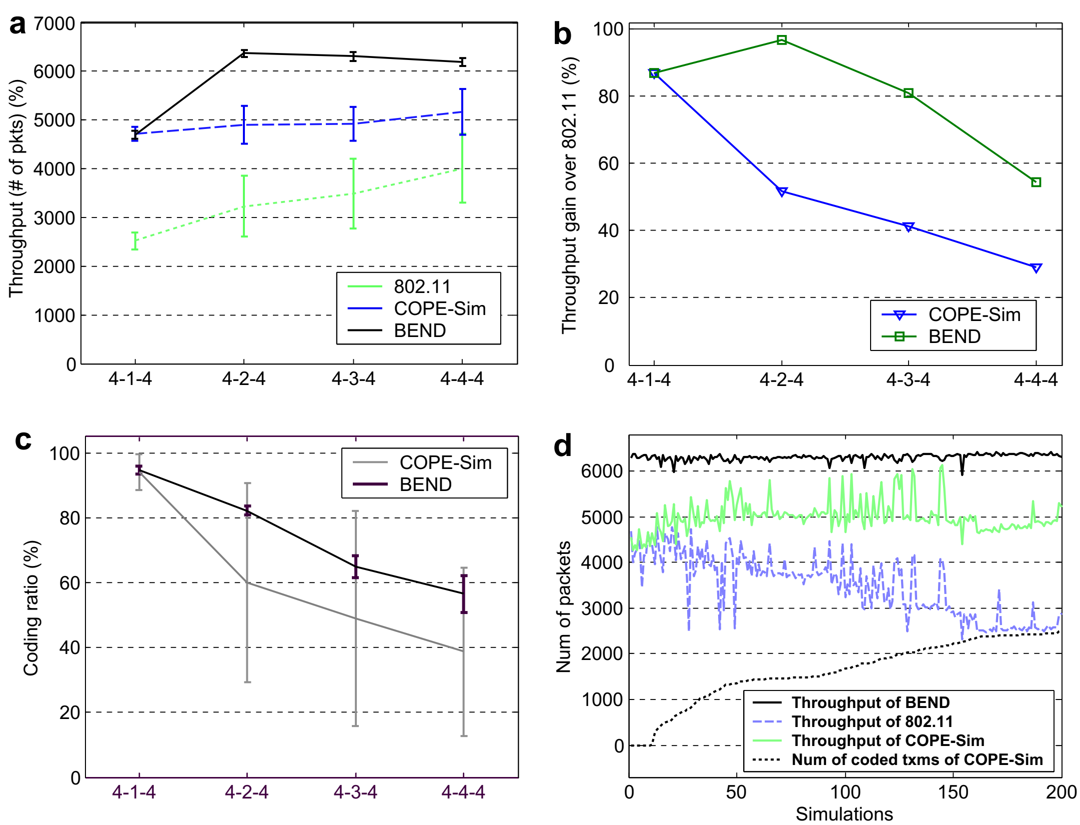


Fig. 7. (a) Throughput of different methods. (b) Throughput gain over 802.11. (c) Coding ratios. (d) Negative correlations between coding gain and diffusion gain.

BEND's performance is not affected by the specific routes determined by the routing module, while COPE-Sim and 802.11 are very sensitive to the level of flow concentration caused by routing. With BEND, diffusion gain and coding gain, which are otherwise conflicting factors, are unified by its power of proactive packet mixing.

BEND solves the dilemma of coding-aware routing and COPE which cannot simultaneously achieve coding gain and diffusion gain. The idea of coding-aware routing is to plan the routes a priori so that the traffic flows are concentrated on certain forwarders. Therefore, for all scenarios coding-aware routing is expected to achieve the same throughput as BEND achieves in 4–1–4 scenario. It is interesting to note in 7a that the throughput achieved by BEND in the scenarios of 4–x–4 (for $x > 1$) is more than 20% higher than 4–1–4. This means that diffusion gain also plays an important role on throughput improvement atop coding gain. However, the traditional fixed-path routing on which the existing coding-aware routing is based cannot exploit coding gain and diffusion gain simultaneously. COPE has the same issue since it uses traditional routing protocols. For COPE-Sim running in presence of multiple tier-2 nodes, either concentrating flows at a particular node provides the coding gain or scattering flows among the forwarders provides the diffusion gain, but *not* both. For example, we take a set of 200 pairs of simulation of 802.11 and COPE-Sim over the 4–3–4 network, each pair records the performance of 802.11 and COPE-Sim using the same routes determined by DSDV. We sort them in the increasing order of the numbers of coded transmissions of COPE-Sim. Then we display them with the throughputs of 802.11, COPE-Sim and BEND in Fig. 7d. Clearly, there is a negative correlation of 802.11's throughput and the number of coded transmissions of COPE-Sim. To the left, where no two reverse flows cross the same tier-2 node, 802.11 achieves its highest throughput but COPE cannot code a single pair of packets. To the right, where all flows cross through a common tier-2 node, 802.11 encounters severe bottleneck effects but COPE-Sim can transmit most packets coded. For BEND, the three tier-2 nodes work as an entity by processing the neighborhood coding repository among themselves, showing a persistently higher throughput gain over COPE-Sim.

5.2. Cross topology

To investigate the capability of coding more than two packets of BEND and COPE-Sim, we design a cross topology of radius 150 m. As a result of the ns-2 settings, each peripheral node has three neighbors (the center and two other orthogonal peripheral nodes) and the center node has four (the peripheral nodes). We place four CBR flows originating from each of the peripheral nodes and terminating at the opposite node. DSDV in this case can pre-

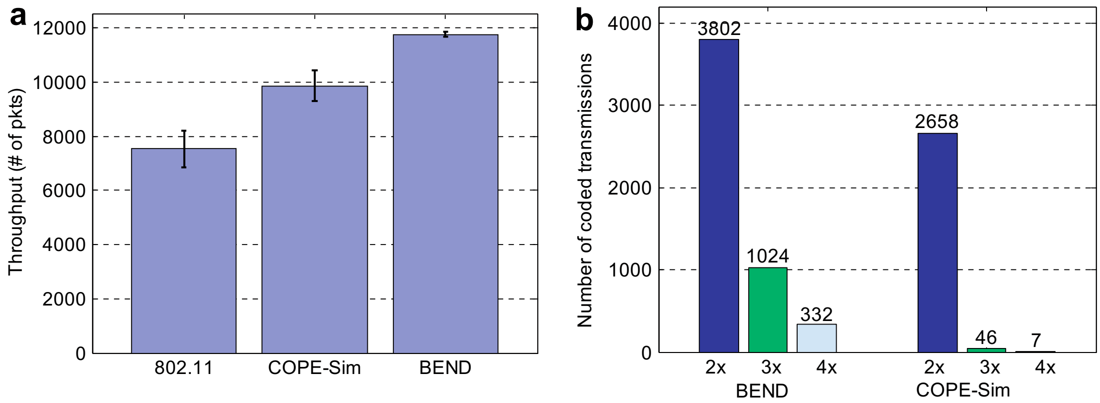


Fig. 8. (a) Throughput and (b) coded transmissions.

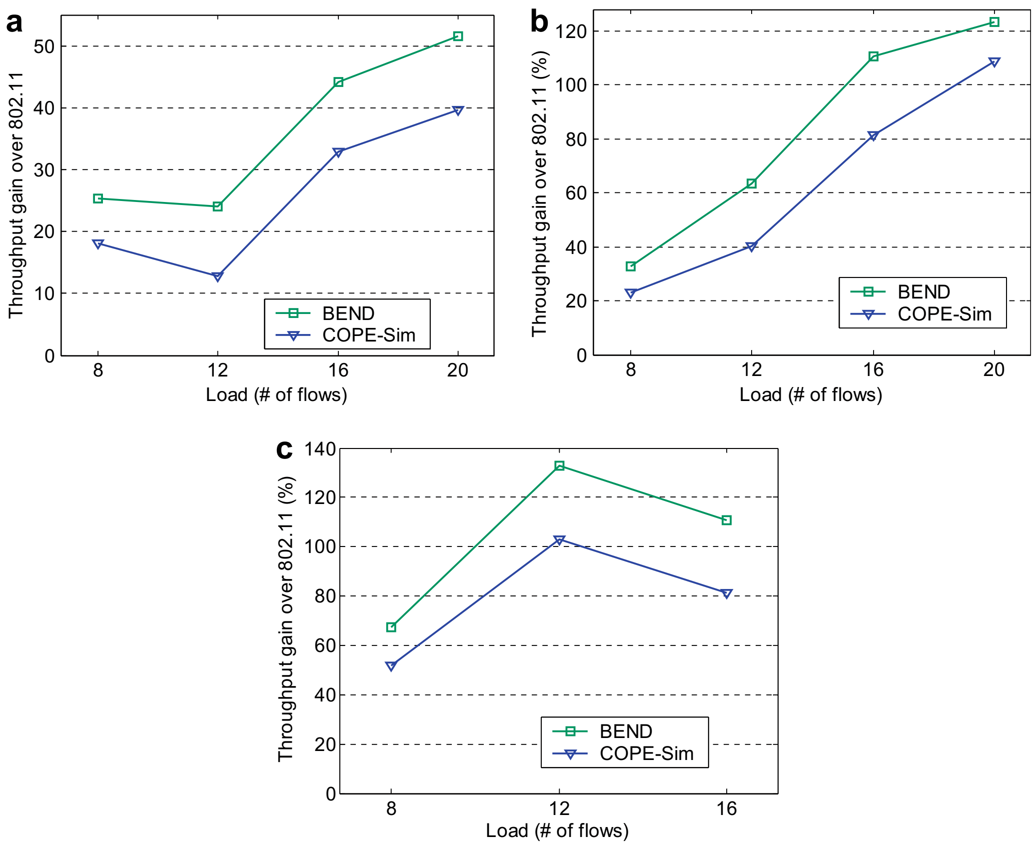


Fig. 9. (a) 2-hop flows; (b) 3-hop flows; and (c) 4-hop flows.

scribe three paths for each flow. In such a configuration, up to four packets can be coded in a single transmission of the center node and up to two packets can be coded together by a peripheral node.

Fig. 8a is the aggregate throughput of the flows supported by 802.11, COPE-Sim and BEND. Again, BEND achieves about double throughput gain relative to 802.11 (56%) compared to COPE-Sim relative to 802.11 (30%), with a much more stable performance.

We have also made the histogram of the number of 2-packet codings, 3-packet codings, and 4-packet codings for BEND and COPE-Sim (Fig. 8b). Apparently, the chance

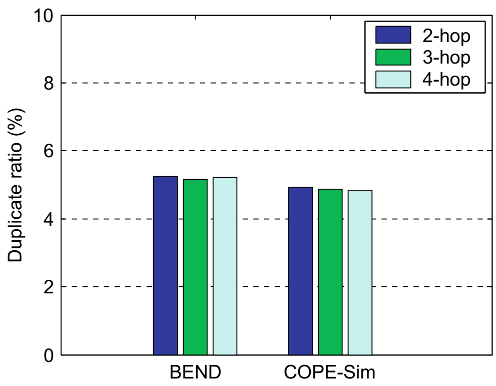


Fig. 10. Duplicate ratio.

that COPE-Sim is able to code more than two packets is very slim given that DSDV will often route more than three flows through a same forwarder. Considering that the four peripheral nodes can only code 2 packets during a single forwarding, the coding of 3 or 4 packets indicates that BEND is very effective in seizing coding opportunities. This is achieved without introducing any artificial backlogging or delay at forwarders although, apparently, holding packets or even pairs or triplets of coded packets in the queue for a bit longer can further boost these numbers. However, we choose not to use this to avoid any delay just for the sake of coding.

5.3. Mesh topology

We further generalize to a 5×5 mesh topology to test the performance of BEND in supporting random flows in larger networks. In the configuration, the grid distances in the two orthogonal directions are set to 150 m. Thus, a non-peripheral node has eight neighbors and a corner node has three neighbors. The diameter of the network is four hops. It is known that multi-hop flows take considerably more network resources to transport the same amount of data [7]. Our goal in this set of experiments is to study BEND's effectiveness with regard to COPE-Sim and 802.11 in supporting a varying number of flows with differing lengths.

We individually test the cases of different flow lengths originating from distinct nodes in the network. For the case of transporting l-hop ($l = 2, 3$, and 4) flows, we vary the number of flows in the network among $f = 8$, 12, 16, and 20. Note that the combination of $l = 4$ and $f = 20$ is impossible given the network diameter of 4 and only 16 peripheral nodes. We plot the throughput of BEND, COPE-Sim and 802.11 for a given hop length in a chart (Fig. 9a–c, respectively). We notice that BEND consistently offers a higher throughput gain than COPE-Sim. In addition, a general trend is that when the number of flows increases, more coding opportunities are found for both BEND and COPE-Sim.

We are also interested in, a larger network, how effective the reliable broadcast and duplication mechanisms are. To measure this, we record the rate of duplicate packets received at UDP receivers for both protocols for each hop length case averaged over all flow numbers tested. As shown in Fig. 10, the duplication rate of BEND is marginally higher than that of COPE-Sim.

6. Discussion and related work

Traditional routing protocols' obliviousness to the coding opportunities was noticed in [13,12,14]. Their solutions focus on routing at the network layer. Such attempts are usually referred to as coding-aware routing. The idea is to compute routes for given flows in a network, taking network coding gain into account, so that the expected total number of transmissions needed to transport the flows is minimized. This is of great importance in theory but the distributed implementation can be rather involved. To compute coding-efficient paths, each node needs to maintain global information of all the flows in the network. The time granularities of traffic lifetime and route update period are usually discrepant. The calculated routes will typically be long dated before being applied to the flows used for the route calculation. More so, due to the extremely close coupling among these flows, any unilateral change of route adopted by an intermediate node will invalidate the purpose of the global routing metric, which is to reduce the number of transmissions. Moreover, the coding-aware routing approach is still based on traditional routing with a single fixed path for each source-destination pair and the redundancy of packets in the network cannot be utilized.

BEND aims at achieving a high coding ratio for each stage of forwarding. It is not globally optimal, but it is flexible, adaptive and practically effective. It only requires local information and the implementation overhead is low. Since it is a per-packet decision for coding, as opposed to per-flow path adaptation, it is more responsive to the dynamics of traffic. BEND also takes advantage of packet redundancy in the network by opportunistic forwarding. The coding chances are greatly improved with multiple potential forwarders instead of one. Moreover, coding-aware routing needs to consider not only the coding gain by combining traffic flows but also their consequential interference. These two forces have been difficult to balance with traditional methods of fixed-path routing. By allowing a set of nodes to process the neighborhood packet repository collectively, BEND adapts to the flow dynamics in the network. On one hand, it proactively mixes data flows that would otherwise go through different nodes in the neighborhood as specified by the routing module. On the other hand, when a specific node happens to be a junction of multiple routes and becomes overloaded, BEND diffuses flows in its neighborhood to alleviate the bottleneck effect. These two aspects are unified under the same framework of BEND. The idea of BEND resembles that of ExOR [2] and MORE [3]. In ExOR, any neighbor en route can forward an overheard data packet as long as it determines that such an opportunistic forwarding leads the packet closer to its destination. Unlike ExOR, which prioritizes forwarders by their distances to the destination, BEND favors those forwarders with a chance to transmit coded packets. ExOR reduces the number of transmissions along the path by skipping some hops if by chance they are received by a node closer to destination. MORE enhances ExOR with network coding to further reduce the transmission redundancy in delivering a single flow from source to destination, i.e. *intra-flow coding*. In contrast, BEND accomplishes efficient delivery by finding more *inter-flow coding* opportunities in the network.

BEND requires no more routing information than what a distance-vector protocol normally offers. In a multi-hop wireless network, the discrepancy among the fragments of routing information maintained at individual nodes of the network can affect the coding decision of BEND to a degree. As a result, there can be occasions where a coded packet cannot be decoded at a receiver because the sender mistakenly decided that decoding was possible using some inconsistent routing information. Even with such minimum requirement for the routing protocol used, BEND can well tolerate such errors and achieve higher data transportation capabilities as shown in the experiments. We believe that the performance of BEND can be further improved if a link-state or a source routing protocol is used where a much more consistent global routing map is stored at each node. However, this is a significantly stronger assumption about the routing module that will compromise the compatibility of BEND with routing protocols.

7. Conclusion and future work

Broadcasting can cause interference in multi-hop wireless networks, but it also brings the benefit of facilitating network coding. When applied effectively, network coding will significantly improve the network's transportation capabilities. The BEND protocol proposed in this paper starts off with the goal of creating more network coding opportunities with a low overhead. It averts the impasse of possibly scarce coding opportunities as with COPE. The key of BEND is to create more coding chances via proactive traffic mixing by treating the packets queued at a neighborhood of forwarders collectively as a distributed packet repository. Our simulation studies indicate that, with minimum assumption on the routing module, BEND consistently achieves higher throughput support than without proactive traffic mixing as in COPE.

The current implementation of BEND makes use of packet redundancy for the encoding aspect. That is, any intermediate node in the neighborhood can encode and forward packets. BEND can be further extended to use packet redundancy for the decoding aspect. In this case, after a node receives a coded packet, if it cannot decode for the non-coded packet intended for it due to some earlier transmission errors, any of its neighbors could decode the packet alternatively and pass the non-coded packet further on to the next hop. One difficulty in realizing this for a distance-vector routing protocol is that this may necessitate the acquisition of the "third-next-hop neighbor" information and including it in the packet header. If a link-state or source routing protocol was adopted instead, this would be a relatively easy extension but would impose a stronger assumption on the routing information. It is also possible to further increase the coding ratio of BEND by introducing more sophisticated delay and scheduling in packets as in [4,15]. This can be another avenue to fine-tune BEND. In this paper, we focus on the throughput performance of UDP flows. It will be interesting to study BEND's capabilities in improving TCP performance. To do that, a packet re-ordering agent needs to be added over BEND so that packets are ordered first before they are passed to the upper-layer TCP receiver to avoid unnecessary TCP congestion backoff. This is a solution employed by ExOR and COPE. In fact, the coordination with TCP in itself is a challenging and important problem for opportunistic routing protocols in multi-hop wireless networks.

References

[1] E. Anderson, T. Anderson, On the stability of adaptive routing in the presence of congestion control, in: Proceedings of the IEEE INFOCOM, 2003.

[2] S. Biswas, R. Morris, Opportunistic routing in multi-hop wireless networks, in: Proceedings of the ACM SIGCOMM, 2005.

[3] S. Chachulski, M. Jennings, S. Katti, D. Katabi, Trading structure for randomness in wireless opportunistic routing, in: Proceedings of the ACM SIGCOMM, 2007.

[4] P. Chaporkar, A. Proutiere, Adaptive network coding and scheduling for maximizing throughput in wireless networks, in: Proceedings of the ACM MobiCom, Montreal, Canada, 2007.

[5] D.S.J. De Couto, D. Aguayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless routing, in: Proceedings of the MobiCom, San Diego, CA, USA, 2003.

[6] C. Fragouli, J.-Y. Le Boudec, J. Widmer, Network coding: an instant primer, ACM SIGCOMM Computer Communication Review 36 (1) (2006) 63–68.

[7] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory 46 (2) (2000) 388–404.

[8] IEEE, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 802.11 Standard, 1997.

[9] IEEE 802.11 WG, Draft Supplement to Standard for Telecommunications and Information Exchange between Systems-LAN/MAN Specific Requirements- Part 11: Wireless LANMAC and Physical Layer (PHY) Specifications: Medium Access Control (MAC), Enhancements for QoS, 802.11e Draft 4.1, February 2003.

[10] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, J. Crowcroft, XORs in the air: practical wireless network coding, in: Proceedings of the ACM SIGCOMM, 2006.

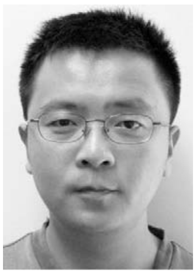
[11] A. Khanna, J. Zinky, The revised ARPANET routing metric, in: Proceedings of the ACM SIGCOMM, 1989.

[12] B. Ni, N. Santhapuri, Z. Zhong, S. Nelakuditi, Routing with opportunistically coded exchanges in wireless mesh networks, Poster Session of IEEE SECON (2006).

[13] S. Sengupta, S. Rayanchu, S. Banerjee, An analysis of wireless network coding for unicast sessions: the case for coding-aware routing, in: Proceedings of the IEEE INFOCOM, 2007.

[14] Y. Wu, S. Das, R. Chandra, Routing with a Markovian metric to promote local mixing, microsoft research technical report, MSR-TR-2006-158, November 2006.

[15] H. Yomo, P. Popovski, Opportunistic scheduling for wireless network coding, in: Proceedings of the IEEE International Conference on Communications (ICC'07), 2007, pp. 5610–5615.



Jian Zhang received his Ph.D. in the Department of Electrical and Computer Engineering at Rutgers University, New Brunswick, NJ. He is now with Integrated Solutions Inc. as a senior Engineer. He received his B.E. in 1994 and M.S. in 1997 from Huazhong University of Science and Technology, China. His research interests include MAC and routing protocols and QoS in wireless mesh/ad hoc networks.

Yuanzhu Peter Chen is an Assistant Professor in the Department of Computer Science at Memorial University of Newfoundland in St. John's, Newfoundland. He received his Ph.D. from Simon Fraser University in 2004 and B.Sc. from Peking University in 1999. Between 2004 and 2005, he was a post-doctoral researcher at Simon Fraser University. His research interests include mobile ad hoc networking, wireless sensor networking, distributed computing, combinatorial optimization and graph theory.

Ivan Marsic received Dipl.Ing. and M.S. degrees in electrical and computer engineering from the University of Zagreb, Croatia, and a Ph.D. degree in biomedical engineering from Rutgers University, New Brunswick, NJ, in 1994. Currently he is an associate professor of electrical and computer engineering at Rutgers University. His research interests include software engineering, computer networks, distributed computing, and human–computer interfaces. His research focuses on engineering complex collaborative systems that include people and computers.