XORs in the Air: Practical Wireless Network Coding

Sachin Katti
Hariharan Rahul
Wenjun Hu
Dina Katabi
Muriel Medard
Jon Crowcroft

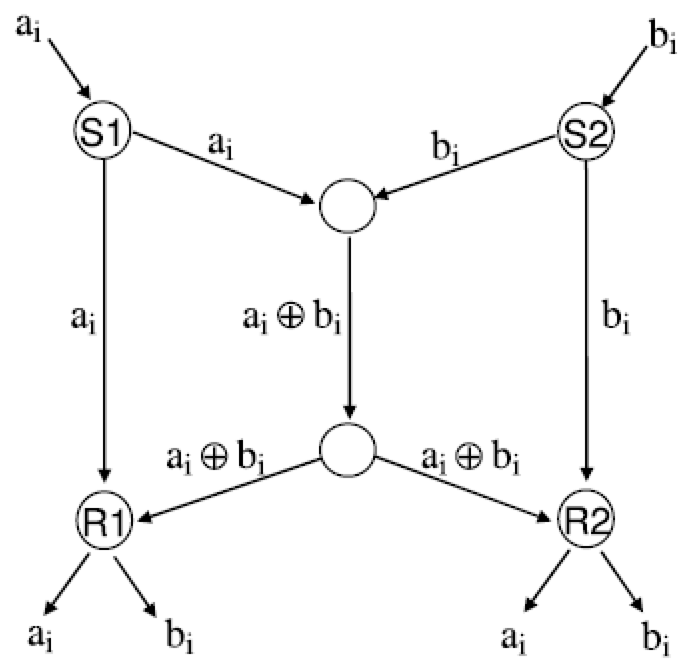
Presented by:
Suvesh Pratapa
suveshp@wpi.edu

Outline

- Background
- COPE – Introduction, Overview
- Understanding COPE's Gains
- Design Issues
- Implementation
- Experimental Results
- Discussion and Conclusion
- Comments

Network Coding – Background

- Ahlswede et al. – *Butterfly Example* in "Network Information Flow", IEEE Transactions on Information Theory, 2000



Allowing routers to mix the bits in forwarding messages can increase network throughput
(Achieves multicast capacity)
This is the basis for Network Coding!

Chronology of Research

- Li et al. – Showed that linear codes are sufficient to achieve maximum capacity bounds (*2003*)
- Koetter and Medard – Polynomial time algorithms for encoding and decoding (*2003*)
- Ho et al. – Extended previous results to a randomized setting (*2003*)
- Studies on wireless network coding began in *2003* as well! (Shows that it was a high interest research area)
- More work on wireless network coding with multicast models (2004)
- Lun et al. – Problem of minimizing communication cost in wireless networks can be formulated linearly (*2005*) – Used multicast model as well!

So all the previous work was theoretical and assumes multicast traffic.

- Authors introduced the idea of opportunistic coding for wireless environments in 2005

Why is it different?

<u>*They address the common case of unicast traffic, bursty flows and other practical issues.*</u>

Current Paper

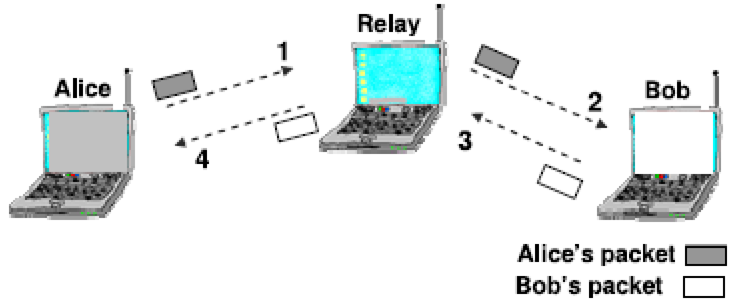
- Explores the utility of network coding in improving the throughput of wireless networks.
- Authors extend the theory of their opportunistic coding architecture (COPE) by application in a practical scenario.
- Presents the first system architecture for wireless network coding.
- Implements the design, creating the first deployment of network coding in a wireless network.
- Studies the performance of COPE.

COPE

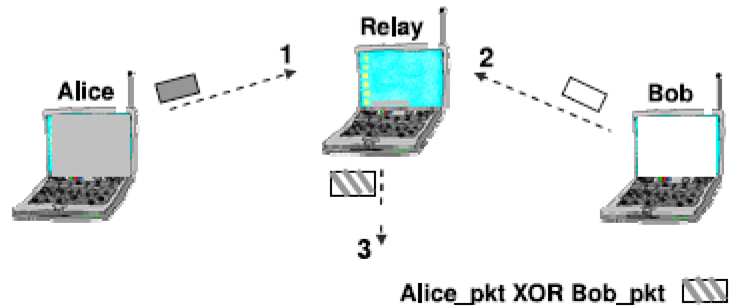
- What does being opportunistic mean?

  Each node relies on local information to detect and exploit coding opportunities when they arise, so as to maximize throughput.

- COPE inserts an *opportunistic* coding shim between the IP and MAC layers.
- Enables forwarding of multiple packets in a single transmission.
- Based on the fact that intelligently mixing packets increases network throughput.



(a) Current Approach



(b) COPE

Design Principles:

- *COPE embraces the broadcast nature of the wireless channel.*
- *COPE employs network coding.*

Inside COPE

Term	Definition
Native Packet	A non-encoded packet
Encoded or XOR-ed Packet	A packet that is the XOR of multiple native packets
Nexthops of an Encoded Packet	The set of nexthops for the native packets XOR-ed to generate the encoded packet
Packet Id	A 32-bit hash of the packet's IP source address and IP sequence number
Output Queue	A FIFO queue at each node, where it keeps the packets it needs to forward
Packet Pool	A buffer where a node stores all packets heard in the past T seconds
Coding Gain	The ratio of the number of transmissions required by the current non-coding approach, to the number of transmissions used by COPE to deliver the same set of packets.
Coding+MAC Gain	The expected throughput gain with COPE when an 802.11 MAC is used, and all nodes are backlogged.

COPE incorporates three main techniques:

- Opportunistic Listening
- Opportunistic Coding
- Learning Neighbor State

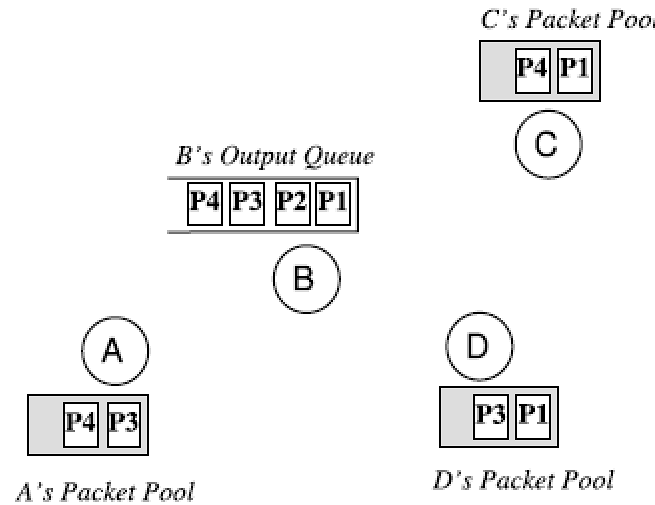
Opportunistic Listening

- Nodes are equipped with omni-directional antennae
- COPE sets the nodes to a promiscuous mode.
- The nodes store the overheard packets for a limited period T (0.5 s)
- Each node also broadcasts *reception reports* to tell it's neighbors which packets it has stored.

Opportunistic Coding

Rule:

"A node should aim to maximize the number of native packets delivered in a single transmission, while ensuring that each intended next-hop has enough information to decode it's native packet."



(a) B can code packets it wants to send

Packets in B's Queue	Next Hop
P1	→ A
P2	→ C
P3	→ C
P4	→ D

(b) Nexthops of packets in B's queue

Coding Option	Is it good?
P1 + P2	Bad Coding (C can decode but A can't)
P1 + P3	Better Coding (Both A and C can decode)
P1 + P3 + P4	Best Coding (Nodes A, C, and D can decode)

(c) Possible coding options

Issues:

- Unneeded data should not be forwarded to areas where there is no interested receiver, wasting capacity.
- The coding algorithm should ensure that all next-hops of an encoded packet can decode their corresponding native packets.

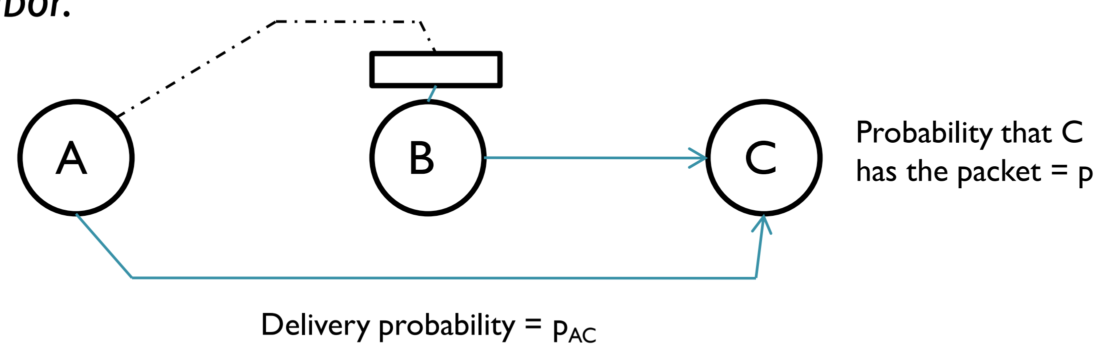
<u>Rule</u>: To transmit n packets $p_1 \ldots p_n$ to n next-hops $r_1 \ldots r_n$, a node can XOR the n packets together only if each next-hop r_i has all n - 1 packets p_j for $j \neq i$

Learning Neighbor State

- A node cannot solely rely on reception reports, and may need to guess whether a neighbor has a particular packet.
- To guess intelligently, we can leverage routing computations.

  The ETX metric computes the delivery probability between nodes and assigns each link a weight of 1/*(delivery_probability)*
- In the absence of deterministic information,

  COPE estimates the probability that a particular neighbor has a packet, as the delivery probability of the link between the packet's previous hop and the neighbor.



"p increases with p_{AC}"

Understanding COPE's Gains

<u>Coding Gain</u>

- Defined as the ratio of no. of transmissions required without COPE to the no. of transmissions used by COPE to deliver the same set of packets.
- By definition, this number is greater than 1.

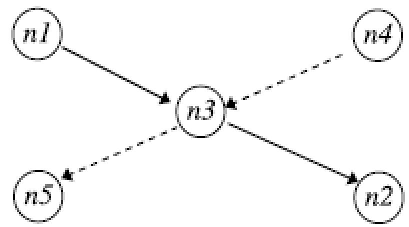
  (4/3 for Alice-Bob Example)
- *<u>Theorem</u>: In the absence of opportunistic listening, COPE's maximum coding gain is 2, and it is achievable.*

Chain topology; 2 flows in reverse directions.

Coding Gain achievable = 2N/(N+1)

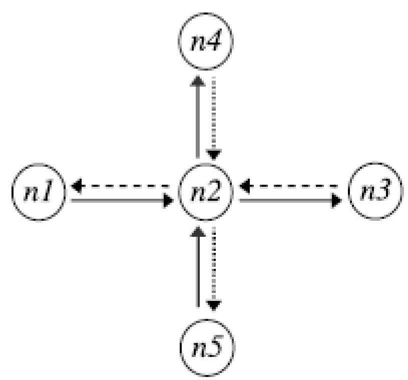
This value tends to 2 as N grows.

In the presence of opportunistic listening



"X" topology
2 flows intersecting at n_2.

Achievable Coding
Gain = 1.33



Cross topology
4 flows intersecting at n_2

Achievable Coding
Gain = 1.6

Understanding COPE's Gains

Coding + MAC Gain

- It was observed that throughput improvement using COPE greatly exceeded the coding gain.
- Since it tries to be fair, the MAC layer divides the bandwidth equally between contending nodes.
- COPE allows the bottleneck nodes to XOR pairs of packets and drain them quicker, increasing the throughput of the network.
- For topologies with a single bottleneck, the Coding + MAC Gain is the ratio if the bottleneck's draining rate with COPE to it's draining rate without COPE.

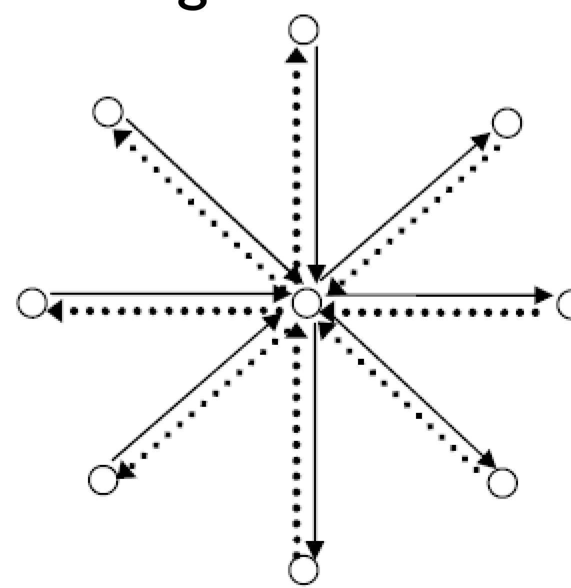
- ***Theorem**: In the absence of opportunistic listening, COPE's maximum Coding + MAC gain is 2, and it is achievable.*

  Node can XOR at most 2 packets together, and the bottleneck can drain at almost twice as fast, bounding the Coding + MAC Gain at 2.

- ***Theorem**: In the presence of opportunistic listening, COPE's maximum Coding + MAC gain is unbounded.*

  For N edge nodes, the bottleneck node XORs N packets together, and the queue drains N times faster.

  The Gain is unbounded.



Wheel topology; many flows intersecting at the center node.

- Theoretical gains:

Topology	Coding Gain	Coding+MAC Gain
Alice-and-Bob	1.33	2
"X"	1.33	2
Cross	1.6	4
Infinite Chain	2	2
Infinite Wheel	2	∞

- Important to note that:
  - The gains in practice tend to be lower due to non-availability of coding opportunities, packet header overheads, medium losses, etc.,
  - But COPE does increase actual information rate of the medium far above the bit rate.

Making it Work – Design Issues

- Packet Coding Algorithm
  - Never delay packets – COPE should not wait for additional codable packets to arrive.
  - Give preference to XORing packets of similar lengths.
  - Never code together packets headed to the same next-hop.
  - Search for appropriate packets to code
  - Packet reordering – Always consider packets according to their order in the queue
  - Ensure that each neighbor to whom packet is headed has a high probability of decoding it's native packet.

$$P_D = P_1 \times P_2 \times \ldots \times P_{n-1}$$

P_D = Probability that the next-hop can decode it's own native packet

P_i = Probability that it has heard packet *I*

(Iterate over the set of neighbors according to a random permutation)

Making it Work

Each node maintains the following data structures:

- Output Queue
- Two per-neighbor virtual queues

  (For small and large packet

  sizes: Threshold = 100)
- Hash table

  (Keyed on *packet-id*)

1 Coding Procedure

```
Pick packet p at the head of the output queue.
Natives = {p}
Nexthops = {nexthop(p)}
if size(p) > 100 bytes then
    which_queue = 1
else
    which_queue = 0
end if
for Neighbor i = 1 to M do
    Pick packet p_i, the head of virtual queue Q(i, which_queue)
    if ∀n ∈ Nexthops ∪ {i}, Pr[n can decode p ⊕ p_i] ≥ G then
        p = p ⊕ p_i
        Natives = Natives ∪ {p_i}
        Nexthops = Nexthops ∪ {i}
    end if
end for
which_queue = !which_queue
for Neighbor i = 1 to M do
    Pick packet p_i, the head of virtual queue Q(i, which_queue)
    if ∀n ∈ Nexthops ∪ {i}, Pr[n can decode p ⊕ p_i] ≥ G then
        p = p ⊕ p_i
        Natives = Natives ∪ {p_i}
        Nexthops = Nexthops ∪ {i}
    end if
end for
return p
```

Making it Work

- Packet Decoding
  - Each node maintains a packet pool
  - When a node receives an XORed collection of packets, it searches for the corresponding native node from it's pool
  - It ultimately XORs the n - 1 packets with the received encoded packet to retrieve it's own native packet.

Making it Work

- Pseudo-Broadcast
  - In 802.11 Unicast, packets are immediately *ack*ed by next-hops and there is an exponential back-off if an *ack* is not received.
  - For 802.11 Broadcast though, since there are many intended receivers, it is unclear who will *ack*. So there are no retransmissions and very low reliability. Throughput is poor.
  - The solution is *Pseudo-Broadcast*.

Making it Work

- Pseudo-Broadcast
  - Piggybacks on 802.11 Unicast

    That means it Unicasts packets meant for Broadcast.
  - Link-layer *dest* field is sent to the MAC address of one of the intended recipients, with an XOR-header added afterward, listing all the next-hops. (All nodes hear this packet)
  - If the recipient receives a packet with a MAC address different from it's own and if it is a next-hop, it processes it further. Else, it stores it in a buffer.
  - Since this is essentially Unicast, collisions are detected, and back-off is possible as well.
  - This does not completely solve the reliability problem.

Making it Work

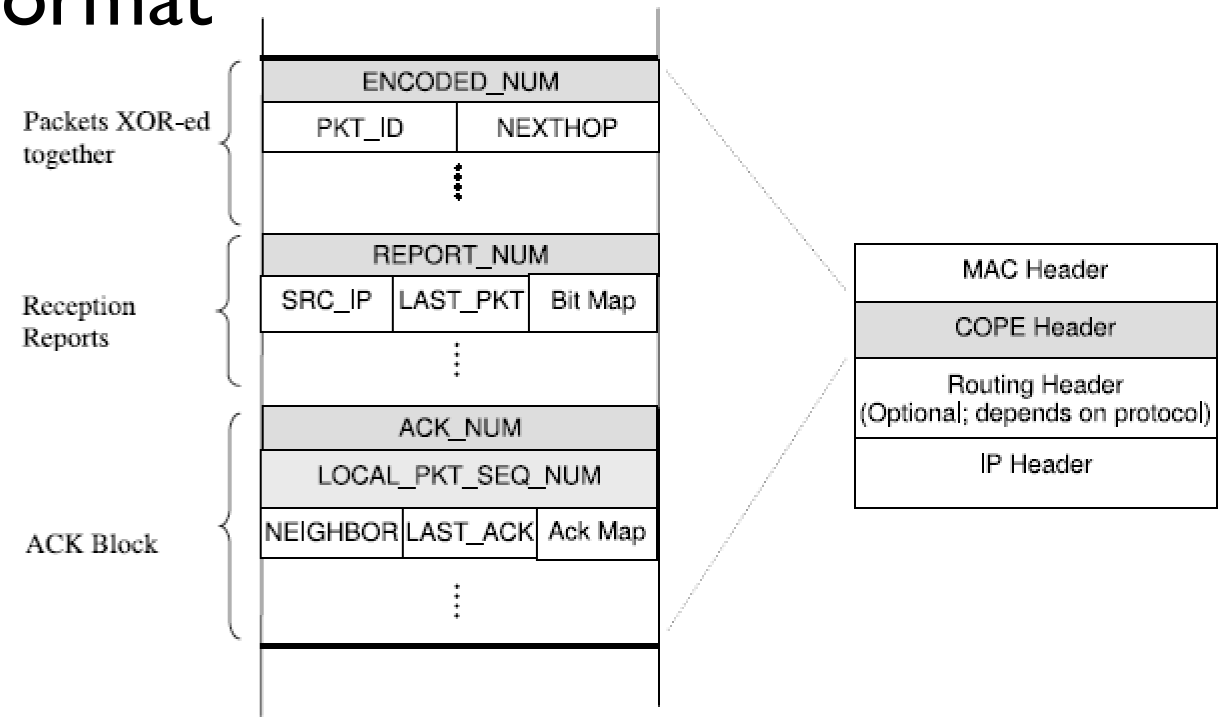
- Hop-by-hop ACKs and Retransmission
  - Probability of loss
    - Not receiving synchronous ACKs.
    - When next-hop actually does not have enough information to decode it's native packet.
  - COPE addresses this problem using local retransmissions.
  - But since there is an overhead with extra headers, encoded packets are *ack*ed asynchronously.
  - Retransmission event is scheduled
  - Next-hop that received an encoded packet also schedules an *ack* event.

Making it Work

- Preventing TCP Reordering
  - Asynchronous *acks* can cause reordering. As mentioned before, reordering can be confused by TCP as a sign of congestion.
  - COPE maintains an *ordering agent*
  - All non-TCP packets and packets whose final IP destinations are different from the current node are taken to the next level.
  - Others are *ordered*! (Using TCP seq numbers)

Implementation

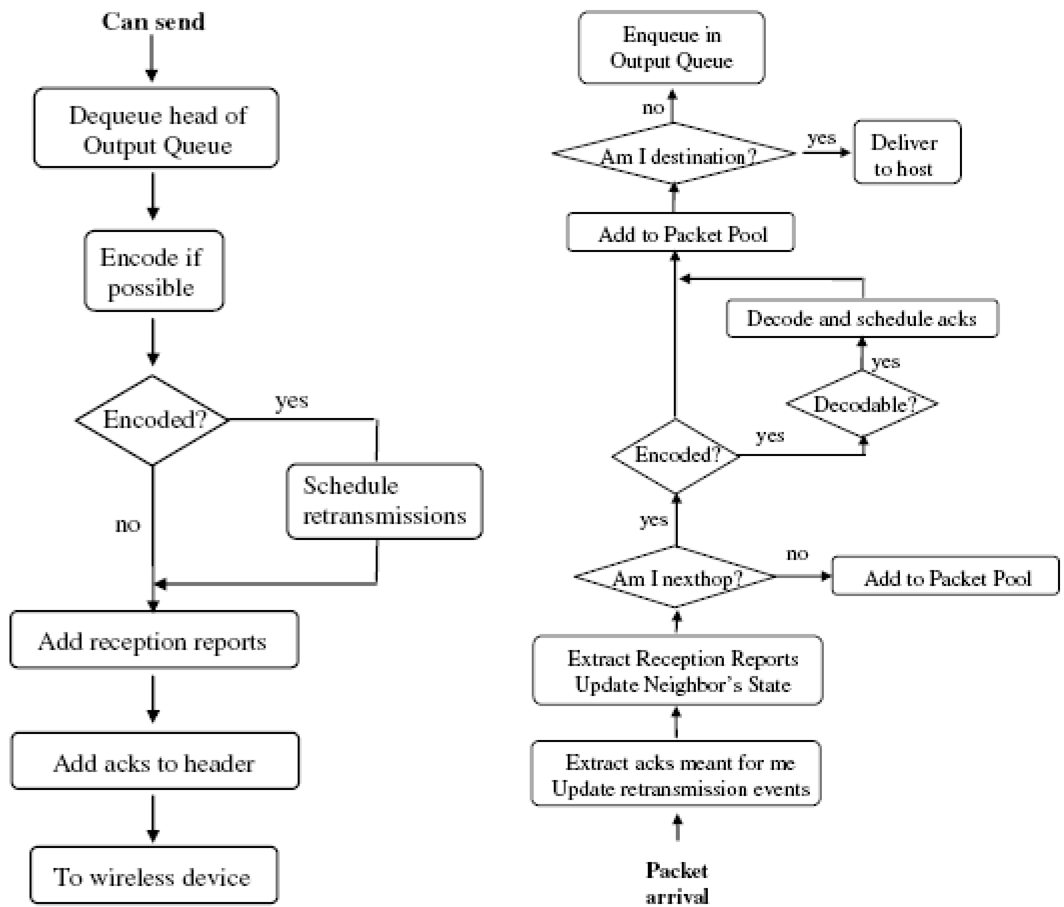
Packet Format



The first block identifies the native packets XOR-ed and their nexthops. The second block contains reception reports. Each report identifies a source, the last IP sequence number received from that source, and a bit-map of most recent packets seen from that source. The third block contains asynchronous acks. Each entry identifies a neighbor, an end point for the ACK map, and a bit-map of ack-ed packets.

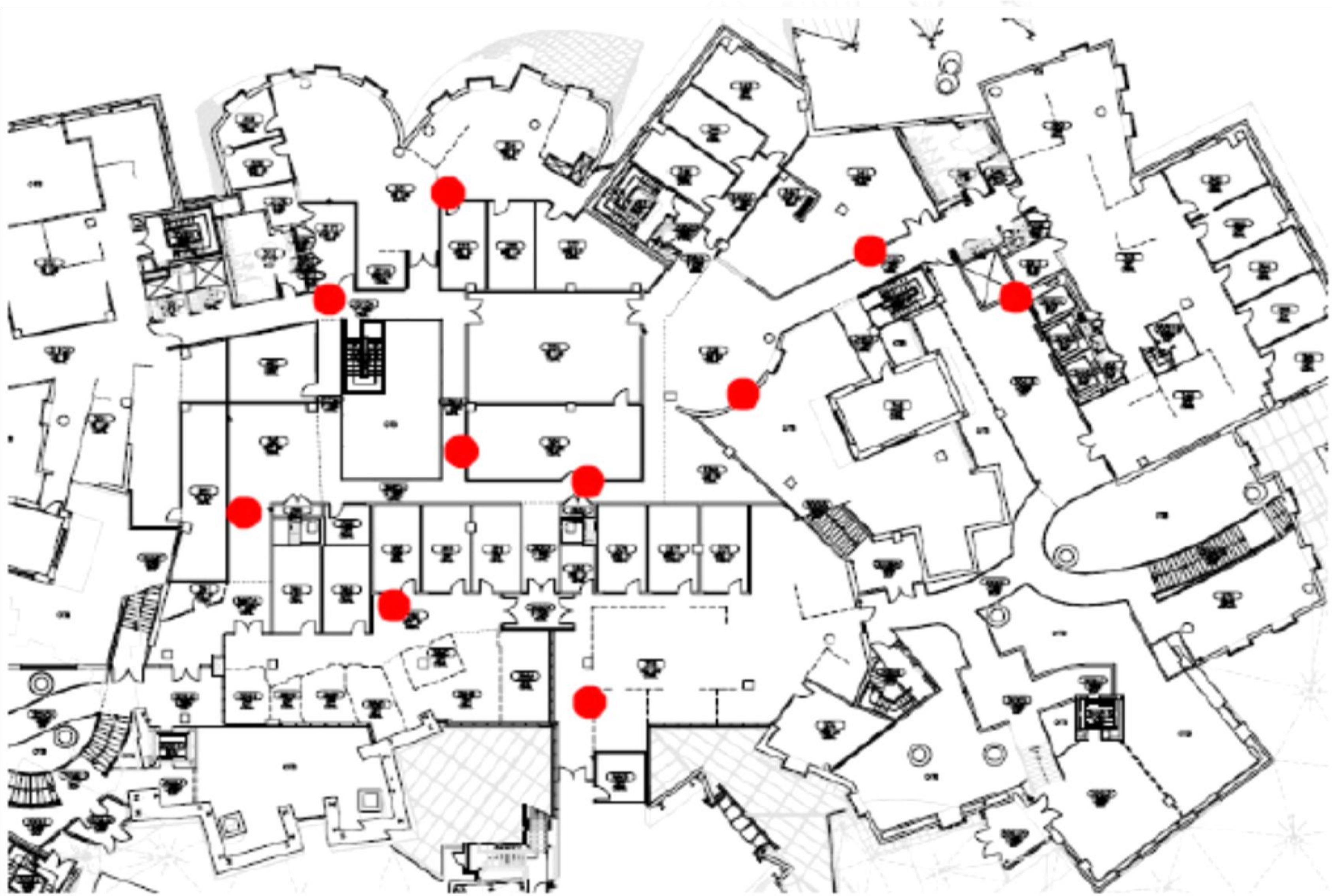
Implementation

Control Flow



(a) Sender side (b) Receiver side

Experimental Results



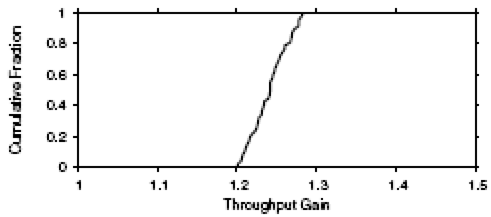
Experimental Results

- Metrics Used
  - Network Throughput (Total end-to-end throughput)
  - Throughput Gain (with and without COPE)
- Three Scenarios
  - COPE in gadget topologies
  - COPE in an Ad Hoc Network
  - COPE in a Mesh Access Network

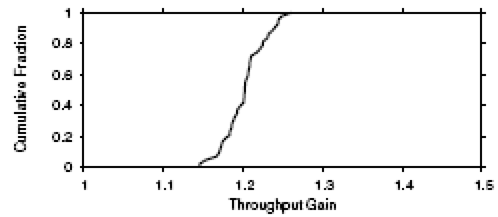
COPE in Gadget Topologies

Study COPE's actual throughput gain (as compared to the theoretical values) using various toy topologies

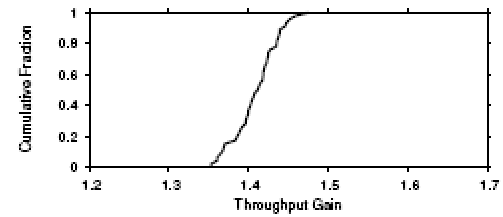
Long-lived TCP Flows



(a) TCP gain in the Alice-and-Bob topology



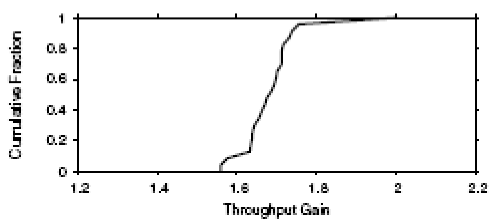
(b) TCP gain in the X-topology



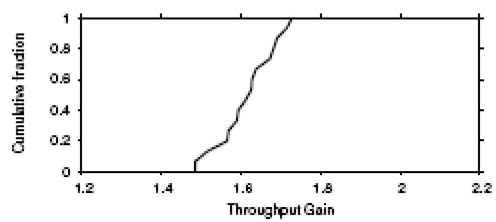
(c) TCP gain in the cross topology

Here, the throughput gain corresponds to only Coding Gain.
Congestion control in TCP balances the draining rate at the bottleneck.

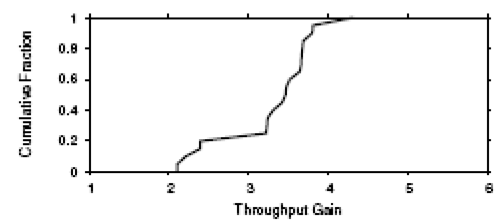
UDP Flows



(a) UDP gain in the Alice-and-Bob topology



(b) UDP gain in the X-topology

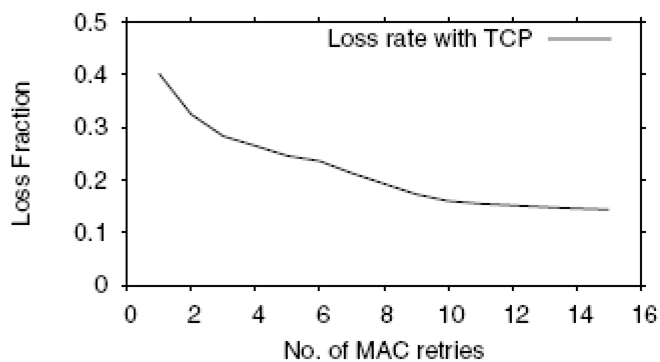


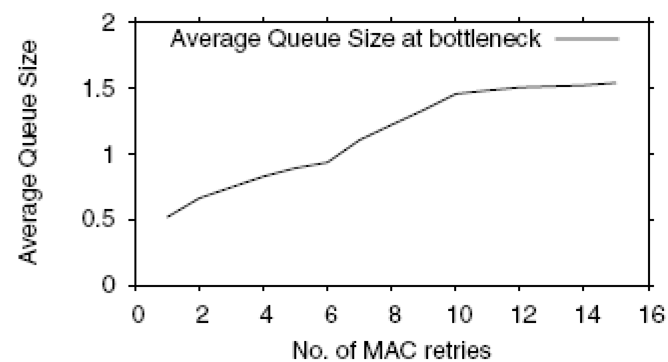
(c) UDP gain in the cross topology

Here, the throughput gain also corresponds to MAC + Coding Gain.
Reduction in throughput is due to XOR header overhead, imperfect overhearing and flow asymmetry.

COPE in an Ad Hoc Network

- TCP flows arrive according to a Poisson process, pick sender and receiver randomly, and the traffic models the Internet.
- TCP does not show significant improvement (2-3%): Collision related losses due to hidden terminals!
- Authors repeat experiment, with varying no. of MAC retries, and with RTS/CTS enabled. COPE is not applied.

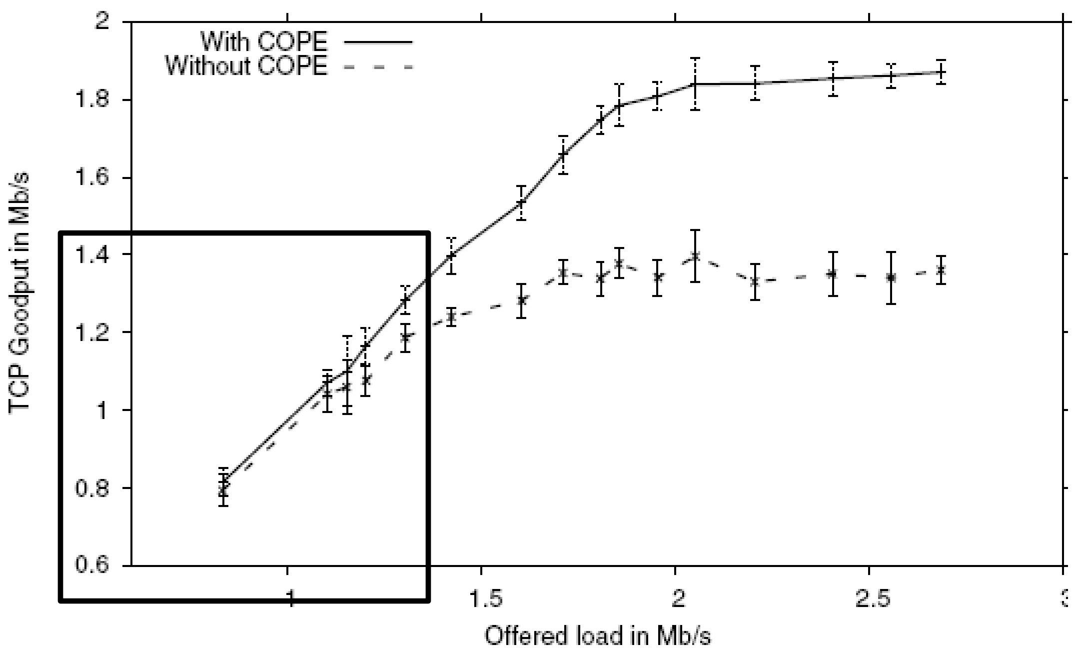


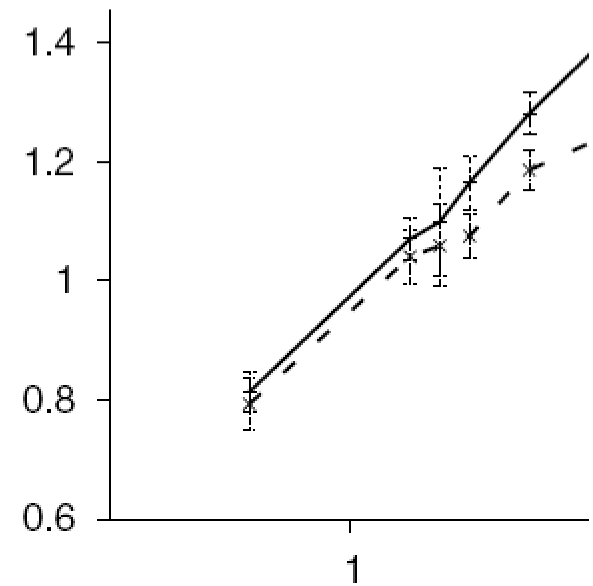


- Even after 15 MAC retries, there is 14% loss, and the bottleneck nodes never see enough traffic. *Few coding opportunities arise!*

COPE in an Ad Hoc Network

- Authors say: *"Making TCP work in collision-related environments would imply solving the problem; but such a solution is beyond the scope of this paper"*
- So prove that it works in a collision-free environment!
- The nodes of the test-bed are brought together, so they are within carrier sense range.

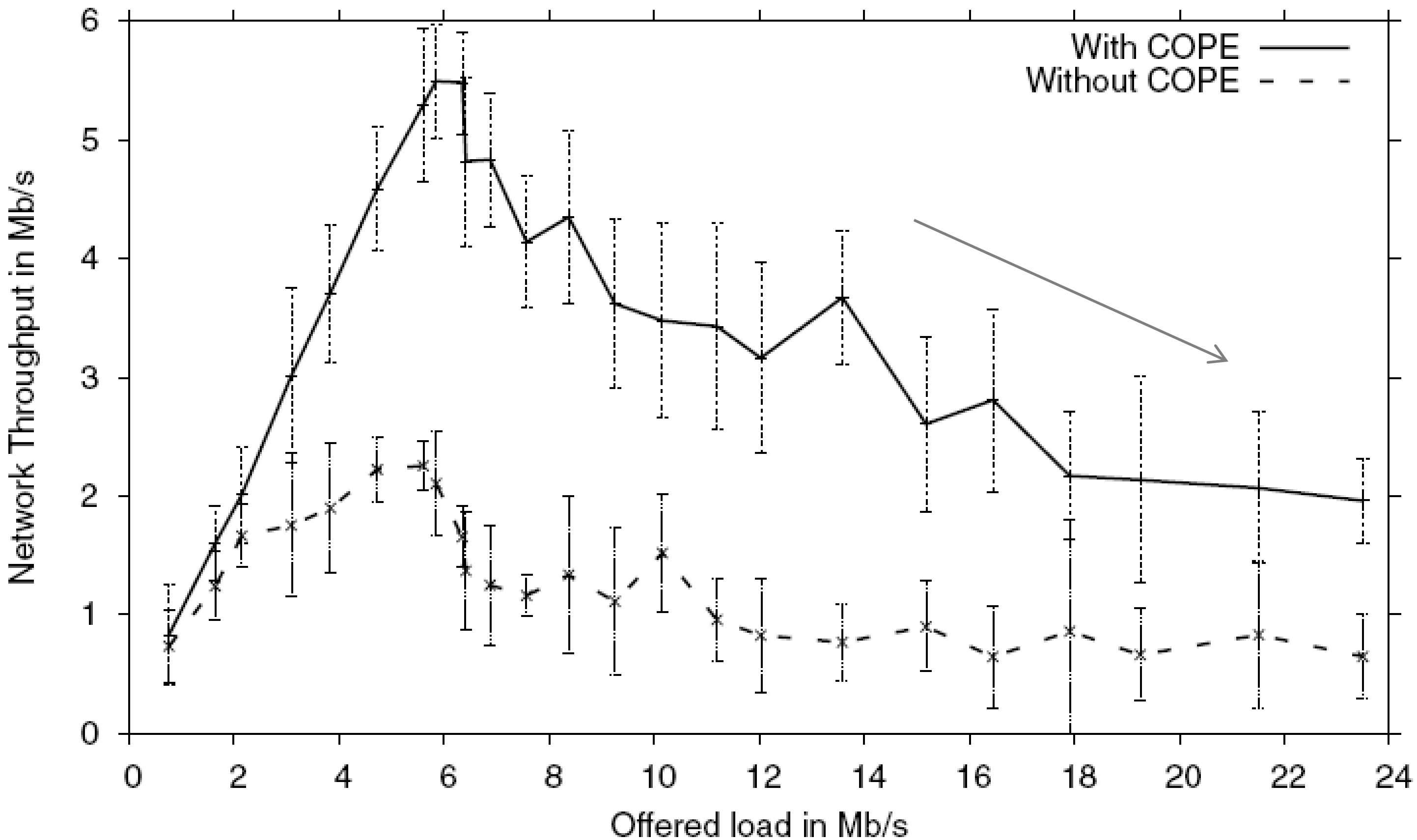




COPE performs well without hidden terminals!

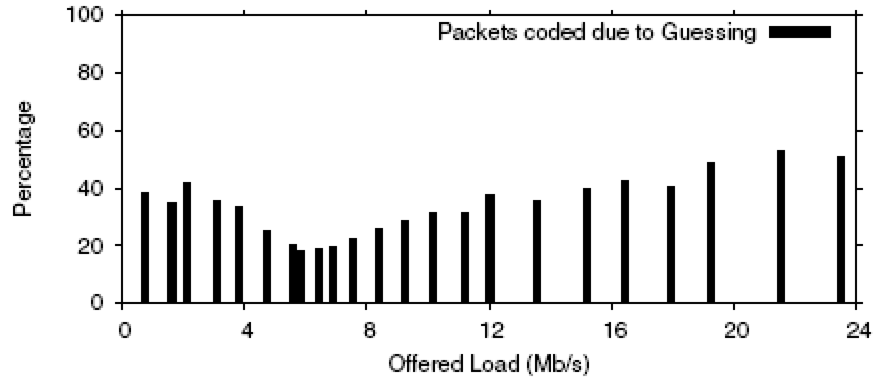
COPE in an Ad Hoc Network

Ok, get UDP into the picture!

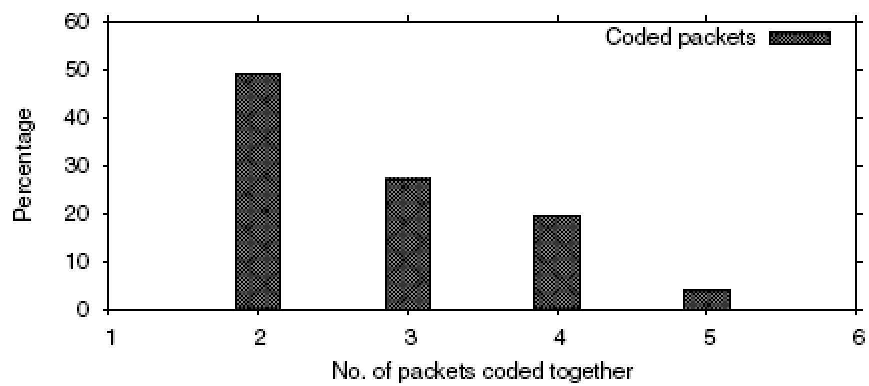


COPE in an Ad Hoc Network

More Observations



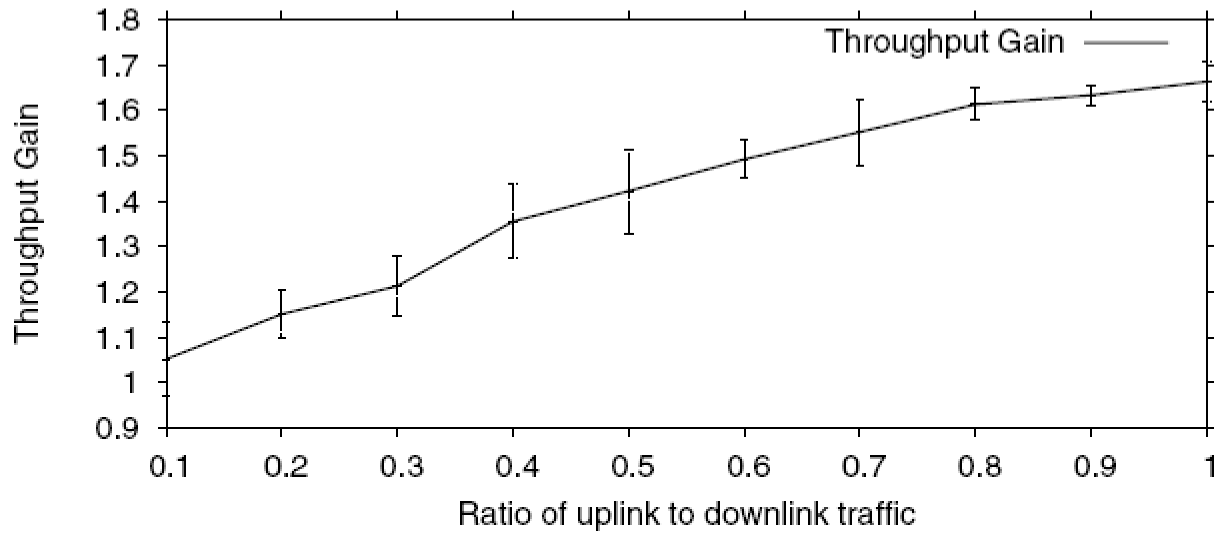
Percentage of packets coded in the testbed due to guessing, as a function of offered load



Distribution of number of packets coded together in the test bed at the peak point

COPE in a Mesh Access Network

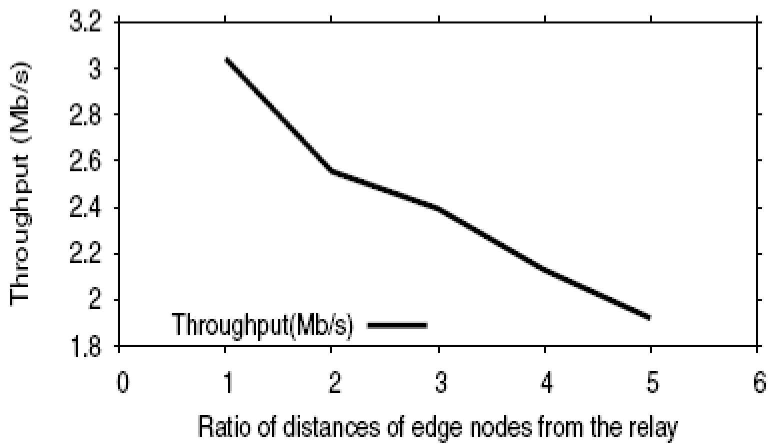
- Multi-hop Wireless Networks that connect to the rest of the Internet via one or more gateways/access points (Traffic flow to and from the closest gateway)
- UDP Flows are used, and uplink/downlink traffic is adjusted.

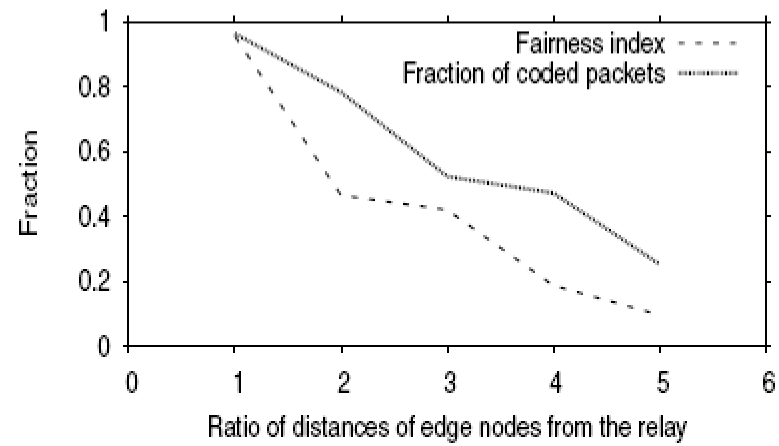


- As the ratio of uplink traffic increases, diversity of the queues at the bottleneck increases, more coding opportunities arise and COPE performs well.

COPE in a Mesh Access Network

- Capture Effect: Sender with better channel captures medium for long intervals.
- Study the effect of capture
- Intentionally stress the links in Alice-Bob topology.





- <u>Result:</u> Without coding, fairness and efficiency conflict with each other. Using coding, these objectives are aligned.

Discussion

- Scope of COPE: Stationary Wireless Mesh Networks
  - Memory: Only packets in flight are used for coding. The storage requirement should be slightly higher than the delay-bandwidth product.
  - Omni-directional antenna: Opportunistic listening exploits the wireless broadcast property.
  - Power requirements: COPE assumes that the nodes are not energy limited.
- COPE can be applied to sensor networks: Nodes can trade-off saved transmissions for reduced battery usage, rather than throughput.
- COPE can be applied to cellular relays: Create a multi-hop cellular backbone with relay nodes to use bandwidth more efficiently. (Ericsson proposed a design where relay XORs only duplex flows)

Conclusion

- Findings:
  - Network Coding does have practical benefits
  - When wireless medium is congested and traffic consists of many random UDP flows, COPE increases throughput by 3 – 4 times.
  - For UDP, COPE's gain exceeds theoretical coding gain.
  - For a mesh access network, throughput improvement with COPE ranges from 5% - 70%
  - COPE does not work well with hidden terminals. Without hidden terminals, TCP's throughput increases by an average of 38%
  - Network Coding is useful for throughput improvement, but COPE introduces coding as a practical tool that can be integrated with forwarding, routing and reliable delivery.

Comments

- No experiments with mixed flows (Briefly mentioned)
- Other routing protocols?
- Should've experimented with 802.11g?
- My overall comment:

  Authors' concept of opportunism is very important because of the broadcast nature of wireless networks – COPE looks to have potential for the future maybe with some tweaks – More sophisticated codes, more compatibility?