15-744: Computer Networking

L-12 Wireless Broadcast

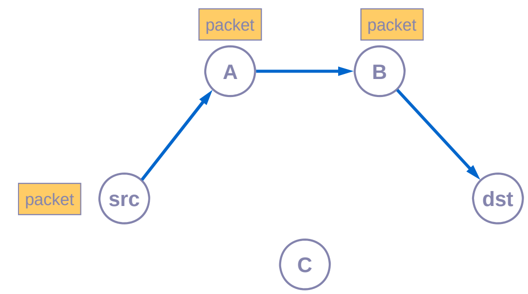
Taking Advantage of Broadcast

- Opportunistic forwarding
- Network coding
- Assigned reading
  - XORs In The Air: Practical Wireless Network Coding
  - ExOR: Opportunistic Multi-Hop Routing for Wireless Networks

Outline

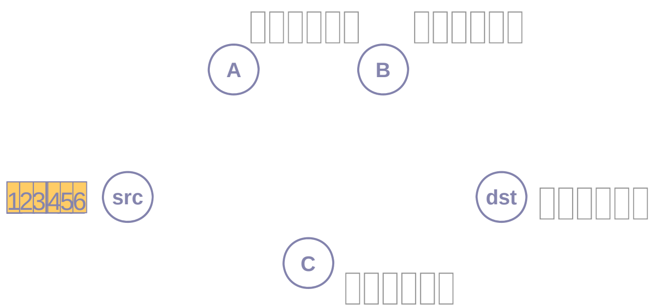
- Opportunistic forwarding (ExOR)
- Network coding (COPE)
- Combining the two (MORE)

Initial Approach: Traditional Routing



- Identify a route, forward over links
- Abstract radio to look like a wired link

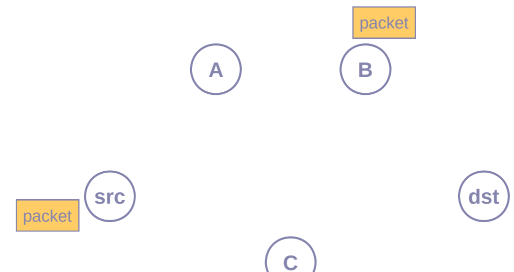
Radios Aren't Wires



- Every packet is broadcast
- Reception is probabilistic

5

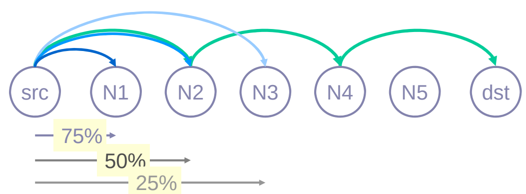
Exploiting Probabilistic Broadcast



- Decide who forwards <u>after</u> reception
- Goal: only closest receiver should forward
- Challenge: agree efficiently and avoid duplicate transmissions

6

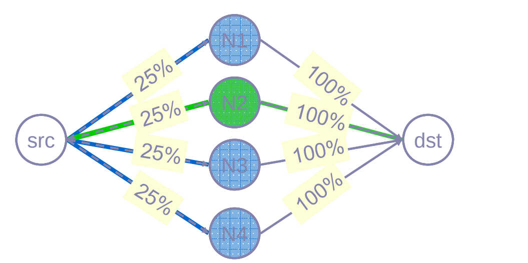
Why ExOR Might Increase Throughput



- Best traditional route over 50% hops: $3(^1/_{0.5}) = 6$ tx
- Throughput $\cong {}^1/_{\#\ \text{transmissions}}$
- ExOR exploits lucky long receptions: 4 transmissions
- Assumes probability falls off gradually with distance

7

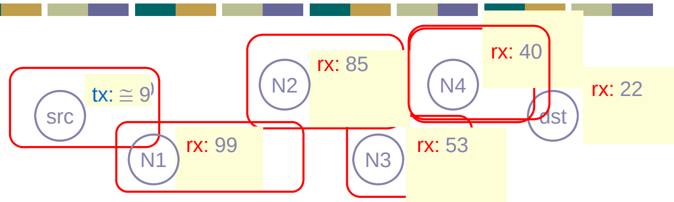
Why ExOR Might Increase Throughput



- Traditional routing: $^1/_{0.25} + 1 = 5$ tx
- ExOR: $^1/_{(1-(1-0.25)^4)} + 1 = 2.5$ transmissions
- Assumes independent losses

8

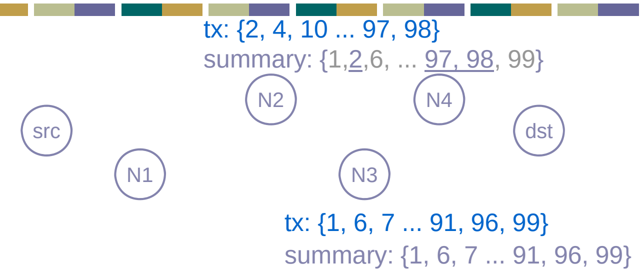
ExOR Batching



- Challenge: finding the closest node to have rx'd
- Send batches of packets for efficiency
- Node closest to the dst sends first
  - Other nodes listen, send remaining packets in turn
- Repeat schedule until dst has whole batch

9

Reliable Summaries



- Repeat summaries in every data packet
- Cumulative: what all previous nodes rx'd
- This is a gossip mechanism for summaries

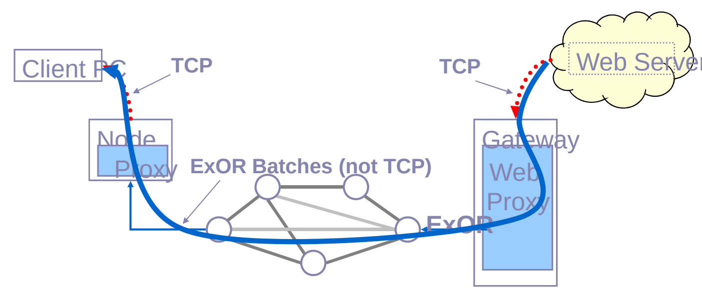
10

Priority Ordering

- Goal: nodes "closest" to the destination send first
- Sort by ETX metric to dst
  - Nodes periodically flood ETX "link state" measurements
  - Path ETX is weighted shortest path (Dijkstra's algorithm)
- Source sorts, includes list in ExOR header

11

Using ExOR with TCP



- Batching requires more packets than typical TCP window

12

Summary

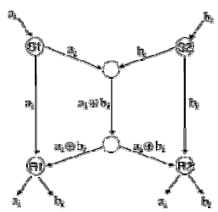
- ExOR achieves 2x throughput improvement
- ExOR implemented on Roofnet
- Exploits radio properties, instead of hiding them

Outline

- Opportunistic forwarding (ExOR)
- Network coding (COPE)
- Combining the two (MORE)

Background

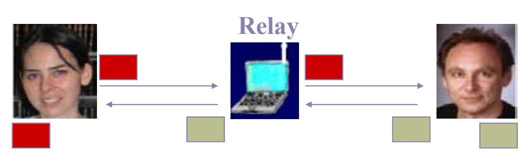
- Famous butterfly example:



- All links can send one message per unit of time
  - Coding increases overall throughput

Background

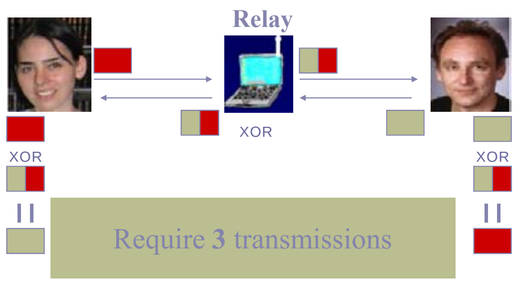
- Bob and Alice



Require 4 transmissions

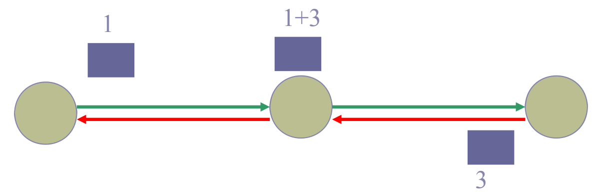
Background

- Bob and Alice



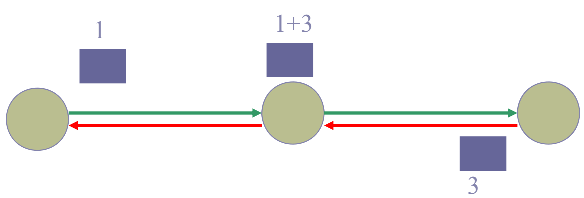
Coding Gain

- Coding gain = 4/3

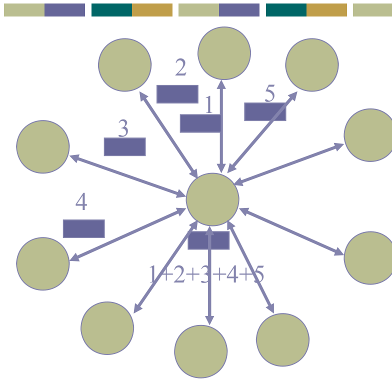


Throughput Improvement

- UDP throughput improvement ~ a factor 2 > 4/3 coding gain



Coding Gain: more examples

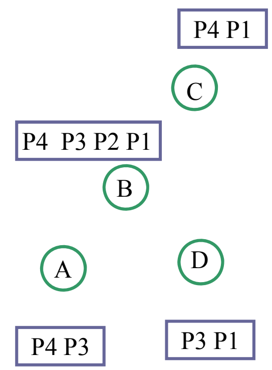


With opportunistic listening, coding gain=2N/(1+N) → 2.
With opportunistic listening, coding gain + MAC gain → ∞

COPE (Coding Opportunistically)

- Overhear neighbors' transmissions
- Store these packets in a ***Packet Pool*** for a short time
- Report the packet pool info. to neighbors
- Determine what packets to code based on the info.
- Send encoded packets

Opportunistic Coding



B's queue	Next hop
P1	A
P2	C
P3	C
P4	D

Coding	Is it good?
P1+P2	Bad (only C can decode)
P1+P3	Better coding (Both A and C can decode)
P1+P3+P4	Best coding (A, C, D can decode)

Packet Coding Algorithm

- When to send?
  - Option 1: delay packets till enough packets to code with
  - Option 2: never delaying packets -- when there's a transmission opportunity, send packet right away
- Which packets to use for XOR?
  - Prefer XOR-ing packets of similar lengths
  - Never code together packets headed to the same next hop
  - Limit packet re-ordering
  - XORing a packet as long as all its nexthops can decode it with a high enough probability

Packet Decoding

- Where to decode?
  - Decode at each intermediate hop
- How to decode?
  - Upon receiving a packet encoded with n native packets
    - find n-1 native packets from its queue
    - XOR these n-1 native packets with the received packet to extract the new packet

Prevent Packet Reordering

- Packet reordering due to async acks degrade TCP performance
- Ordering agent
  - Deliver in-sequence packets immediately
  - Order the packets until the gap in seq. no is filled or timer expires

Summary of Results

- Improve UDP throughput by a factor of 3-4
- Improve TCP by
  - wo/ hidden terminal: up to 38% improvement
  - w/ hidden terminal and high loss: little improvement
- Improvement is largest when uplink to downlink has similar traffic
- Interesting follow-on work using analog coding

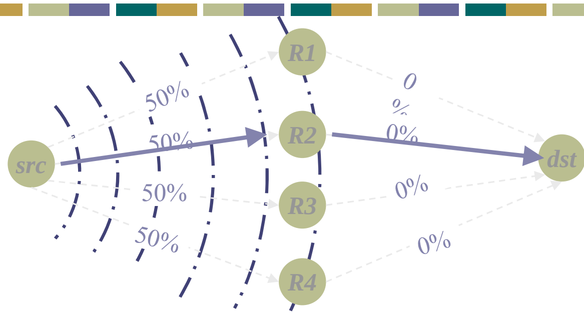
Reasons for Lower Improvement in TCP

- COPE introduces packet re-ordering
- Router queue is small ➔ smaller coding opportunity
  - TCP congestion window does not sufficiently open up due to wireless losses
- TCP doesn't provide fair allocation across different flows

Outline

- Opportunistic forwarding (ExOR)
- Network coding (COPE)
- Combining the two (MORE)

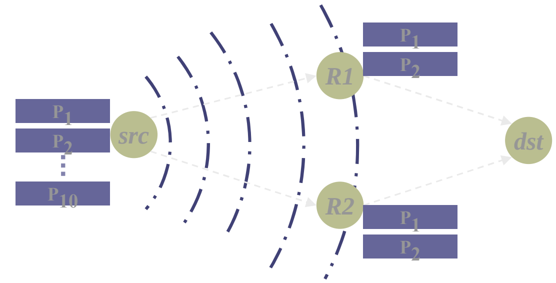
Use Opportunistic Routing



Opportunistic routing promises large increase in throughput

But

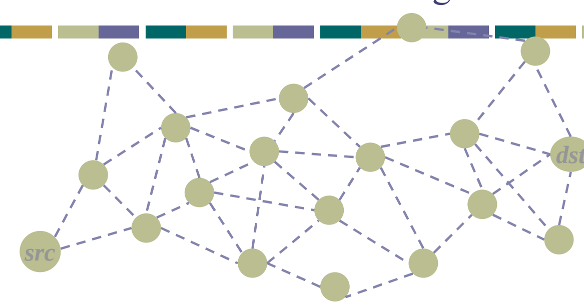
- Overlap in received packets → Routers forward duplicates



ExOR

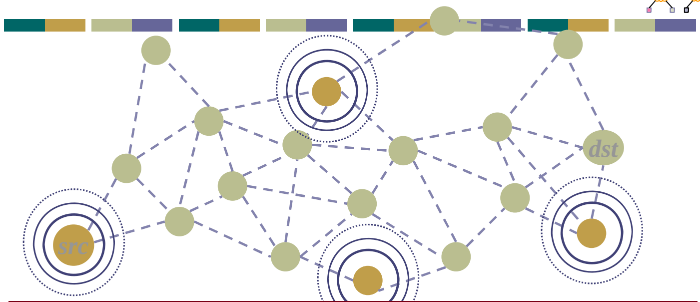
- State-of-the-art opp. routing, ExOR imposes a global scheduler:
- Requires full coordination; every node must know who received what
- Only one node transmits at a time, others listen

Global Scheduling?



- Global coordination is too hard
- One transmitter

Global Scheduling?



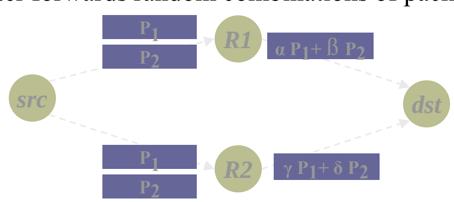
Does opportunistic routing
have to be so complicated?

MORE (Sigcomm07)

- Opportunistic routing with no global scheduler and no coordination
- We use random network coding
- Experiments show that randomness outperforms both current routing and ExOR

Go Random

Each router forwards random combinations of packets

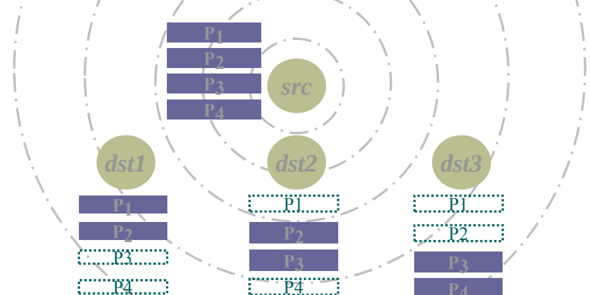


Randomness prevents duplicates

No scheduler; No coordination

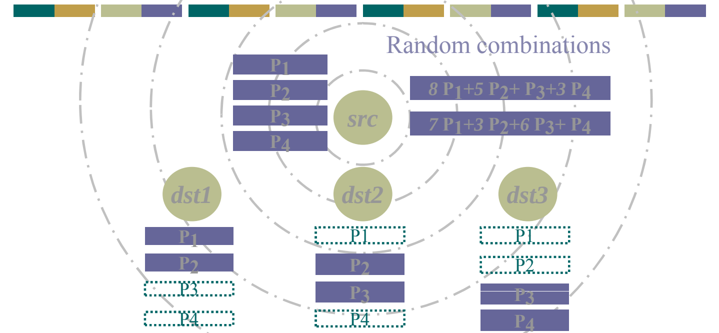
Simple and exploits spatial reuse

Random Coding Benefits Multicast



Without coding → source retransmits all 4 packets

Random Coding Benefits Multicast

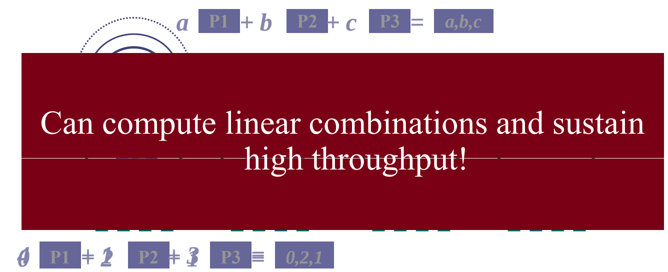


Random coding is more efficient than global coordination

37

MORE

- Source sends packets in batches
- Forwarders keep all heard packets in a buffer
- Nodes transmit linear combinations of buffered packets

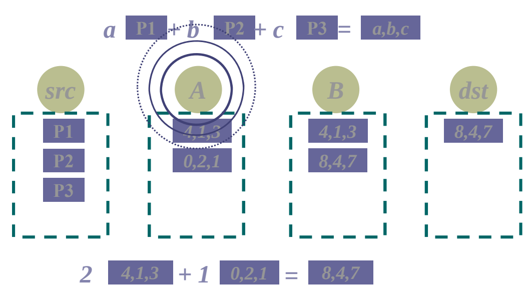


Can compute linear combinations and sustain high throughput!

38

MORE

- Source sends packets in batches
- Forwarders keep all heard packets in a buffer
- Nodes transmit linear combinations of buffered packets



39

MORE

- Source sends packets in batches
- Forwarders keep all heard packets in a buffer
- Nodes transmit linear combinations of buffered packets

- Destination decodes once it receives enough combinations
  - Say batch is 3 packets

1 P1 + 3 P2 + 2 P3 = 1,3,2

5 P1 + 4 P2 + 5 P3 = 5,4,5

4 P1 + 5 P2 + 5 P3 = 4,5,5

- Destination acks batch, and source moves to next batch

40