Trading Coordination For Randomness

Szymon Chachulski

Mike Jennings, Sachin Katti, and Dina Katabi
Wireless mesh networks have high loss rates

**Objective:**
High throughput despite lossy links

*Roofnet*

*Avg. 30% loss*
Use Opportunistic Routing
Use Opportunistic Routing

- Best single path $\rightarrow$ loss prob. 50%
Use Opportunistic Routing

Opportunistic routing promises large increase in throughput
But

Overlap in received packets $\rightarrow$ Routers forward duplicates
But

Overlap in received packets $\rightarrow$ Routers forward duplicates
But

Overlap in received packets → Routers forward duplicates

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?

Can we do opportunistic routing without global scheduling?
Contributions

• 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
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 combinations

8 P1+5 P2+ P3+3 P4
7 P1+3 P2+6 P3+ P4

Random coding is more efficient than global coordination
MORE

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

\[ a \cdot P_1 + b \cdot P_2 + c \cdot P_3 = a, b, c \]

Can compute linear combinations and sustain high throughput!

\[ P_1 \neq 1 \quad P_2 \neq 2 \quad P_3 = 0, 2, 1 \]
MORE

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

\[ aP_1 + b + cP_2 + c = a,b,c \]

\[ 2 \cdot P_1 + P_2 + P_3 = a,b,c \]

\[ \begin{align*}
2 & \cdot P_1 + 1 \cdot P_2 + 0,2,1 = 8,4,7 \\
8,4,7 & = 8,4,7
\end{align*} \]
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

\[
\begin{align*}
1 & \quad P1 + 3 \quad P2 + 2 \quad P3 = 1,3,2 \\
5 & \quad P1 + 4 \quad P2 + 5 \quad P3 = 5,4,5 \\
4 & \quad P1 + 5 \quad P2 + 5 \quad P3 = 4,5,5
\end{align*}
\]

• Destination acks batch, and source moves to next batch
But How Do We Get the Most Throughput?

- Naïve approach transmits whenever 802.11 allows

If $A$ and $B$ have same information, it is more efficient for $B$ to send it

Need a method to determine how much each node should send.
Probabilistic Forwarding
Probabilistic Forwarding

Loss rate 0%

Loss rate 50%
Probabilistic Forwarding

How many packets should I forward?

50% of buffer
Probabilistic Forwarding

Compute forwarding probabilities without coordination using loss rates.
Can ExOR Use Probabilistic Forwarding To Remove Coordination?

• Without random coding $\rightarrow$ need to know the **exact** packets to forward every time

• With random coding $\rightarrow$ need to know only the **average** amount of overlap

Probability of duplicates is 50%
Adapting to Short-term Dynamics

• Need to balance sent information with received information

• MORE triggers transmission by receptions

• A node has a credit counter
  □ Upon reception, increment the counter using forwarding probabilities
  □ Upon transmission, decrement the counter

• Source stops → No triggers → Flow is done
Performance
Experimental Setup

• We implemented MORE in Linux

• 20-node testbed

• Compare MORE with:
  □ Current Routing (Single Best Path)
  □ ExOR (State-of-the-art Opportunistic Routing)

• Experiment
  □ Random source-destination pairs
  □ Transmit 5 MB file
Testbed

- 20-node testbed over three floors
Testbed

• 20-node testbed over three floors

Avg. loss 27%
Does MORE Improve Wireless Throughput?

MORE’s throughput is

- 2x better than current routing
- 22% better than ExOR
Throughput of All Source-Destination Pairs

CDF of 180 source-destination pairs

Throughput [packets/s]

Current
ExOR
MORE
Zoom in on the worst 10%

MORE addresses dead spots

Current
ExOR
MORE

4x
Sensitivity to Batch Size

ExOR CDF

MORE CDF

Batch = 8 pkts
Batch = 128 pkts

MORE works for short flows
What About Multicast?

MORE improves both multicast and unicast throughput

 Avg. Throughput Per Destination [pkts/s]

<table>
<thead>
<tr>
<th>Destination</th>
<th>Current</th>
<th>MORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>+200%</td>
<td>+260%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>+320%</td>
</tr>
</tbody>
</table>

MORE improves both multicast and unicast throughput.
MORE for Less!

• Less coordination and less rigidity
  □ No global scheduler

• More flexibility
  □ Works on top of 802.11 → enjoy spatial reuse
  □ One framework for unicast and multicast

• More throughput
  □ 22% better than ExOR
  □ 2x better than current routing
Comments?
Comments

• Cool idea to combine network coding with opportunistic routing

• Works for multiple flows and multicast!
  □ Performance under more flows is not good

• Performance improvement over ExOR is marginal – 25%

• Performance issues?
Comments

• Cool idea to combine network coding with opportunistic routing

• Works for multiple flows and multicast!
  □ Performance under more flows is not good

• Performance improvement over ExOR is marginal - 25%

• Performance issues?
  □ Stop-and-wait
  □ End-to-end recovery
  □ Lack of rate limit
Review

• What is opportunistic routing?
• What are its benefits?
• What is the main challenge?
• How does ExOR address the challenge?
• How does MORE address the challenge?
• Pros & Cons?