Multiple Network Coded TCP Sessions in Disruptive Wireless Scenarios

Chien-Chia Chen
Cliff Chen
Joon-Sang Park
Soon Oh
Mario Gerla
M.Y. Sanadidi

Network Research Lab
CSD, UCLA
Problem Statement—Communication over Disruptive Networks

- Applications (streaming, TCP) mostly do not work
- Sample Scenario
  - Military environment
    - Jamming, obstacles, and mobility in combat scenario
  - Civil environment
    - Obstacles and mobility in urban area
TCP Scenario

- Problems

1. **random** losses are misinterpreted as **congestion**
2. TCP DATA and ACK flows **contend** for the same shared medium
TCP over Unstable Wireless Links

TCP-DATA
Source

54Mbps

Destination

TCP-ACK

Throughput (bps)

Simulation Time (Sec)

0%PER

40%PER

20%PER
TCP Previous Work—Intra-Flow Coding

- To mitigate high error rate, intra-flow coding is a known approach
  - ✓ Uses Random Linear Coding to recover all losses
  - ✗ TCP modification on both sender and receiver
  - ✗ Uncontrolled redundancy
  - ✗ Does not address TCP DATA-ACK interference

![Diagram showing TCP DATA and TCP ACK with random loss and sources and destinations]
TCP Previous Work—Inter-Flow Coding

- To relieve TCP DATA-ACK interference
  - XOR-based network coding
  - PiggyCode
    - ✓ Opportunistically XOR DATA and ACK at relays
    - ❌ Not robust to random losses
    - ❌ Requires MAC layer modifications
TCP Scenario—Proposed Solution

- To mitigate high loss → **Intra-Flow Coding**
- To mitigate DATA-ACK interference → **Inter-Flow Coding**
- Transparent to Upper/Lower Layers
Adapt to Varying Losses

- Link error rates are changing at all times

- Each node stamps “number of received packet” in packets header
  - Upstream node receives it
  - It adjusts link coding redundancy based on successful delivery (to the next hop)
Redundancy Control

- $N_{i+1}$: packets received at node $i+1$ in current generation
- $M_i$: packets sent from node $i$ in current generation
- **instantaneous loss** $P_i^0 = \frac{M_i - N_{i+1}}{M_i}$
- **exponential average** $\overline{P}_i' = \overline{P}_i + \alpha \times \left( \overline{P}_i^0 - \overline{P}_i \right)$
- **coding redundancy** $R_i = (K_i - 1) + \frac{1}{1 - \overline{P}_i'}$

where $K_i$ is base redundancy (1.6 in the simulation)
Simulation Configurations

- 802.11g Unicast
  - CSMA/CA
  - RTS/CTS is DISABLED
  - MAC ACK and MAC retransmission (up to 4 times)
  - Promiscuous Mode ENABLED

- Traffic: FTP/TCP-NewReno
- Relays re-encode coded TCP-DATA packets
- Experimentally optimized Coding Redundancy
  - # of coded packets / # of original packets
  - 1.4~2.0 based on packet loss rate
More Parameters

- Topology: 3-hop string
- 4 Sets of Simulations
  - TCP-NewReno (without coding help)
  - PiggyCode (Inter-Flow Coding) (timer=4ms)
  - Pipeline Coding (Intra-Flow Coding) (with experimentally optimal redundancy)
  - ComboCoding (with the above experimentally optimal setting)
- Generation size for random linear coding is 16
Link Error Rates

- Vary per link Packet Error Rate over time
  - 20~50 sec: 0% PER
  - 50~80 sec: 40% PER
  - 80~110 sec: 20% PER

- Results (measuring goodput)
  - Under perfect links, PiggyCode is the best
  - Under unstable lossy links, adaptive control helps
  - Redundancy Controlled ComboCoding is the most stable
Simulation Results—No Redundancy Control
X Topology Simulation Setup

- 2 TCP Flows
- 802.11g (CSMA/CA, NO RTS/CTS): 54Mbps
- Gen size: 16
Goodput (No Coding vs. ComboCoding)
Goodput (PiggyCode vs. Pipeline Coding)
Fairness (Jain’s Index)

The diagram shows the fairness (Jain’s Index) over simulation time for different TCP coding schemes. The x-axis represents simulation time in seconds, while the y-axis represents the fairness index. The legend indicates five different coding schemes:

- TCP No Coding
- TCP PiggyCode
- TCP ComboCoding
- TCP Pipeline Coding

The graph includes markers for 0% PER, 40% PER, and 20% PER, indicating different error rates in the simulation.
Grid Topology
System Goodput

The graph shows the system goodput over time for different TCP coding strategies. The x-axis represents the simulation time in seconds, ranging from 20 to 110. The y-axis represents the goodput in Mpps, ranging from 0 to 10. Different lines represent various TCP coding schemes:

- Green dots: TCP No Coding
- Green triangles: TCP PiggyCode
- Blue crosses: TCP ComboCoding
- Light blue plus signs: TCP Pipeline Coding

The graph is marked with PER values: 0%PER, 40%PER, and 20%PER, indicating different error rates in the simulation.
Fairness (Jain’s Index)
Conclusion

- **ComboCoding** provides an efficient and robust coding scheme for TCP
- However, still work remained to be studied
  - Adaptive Redundancy Control
  - Testbed Evaluation
THANK YOU 😊