# CSE 222
# Graduate Networking

## Winter 2001

Lecture 16: Wireless Networks

Geoffrey M. Voelker

## Wireless Networking

- Many topics in wireless networking
  - Transport optimizations, ad hoc routing, MAC algorithms, QoS, mobility, etc.
- Survey papers on TCP optimizations, ad hoc routing
  - Good mileage – cover a lot of ground in just two papers
  - Can often understand the problem and issues better when different approaches are compared and contrasted
- These papers reflect one goal of the course
  - Pick up a paper like this and understand what all of the buzzwords mean…
    - » Reno, NewReno, SACK, distance vector, link state, etc.
  - Look, ns!

## Comparing Approaches

- These two papers are also good examples of methodologies for comparing approaches to problems
  - If you're in a similar situation, think back to these papers
- What's involved?
  - Likely will have to implement approaches yourself
    - » Requires good understanding of all approaches
      - Sometimes better than original developers!
    - » A lot of work, both implementing and performing experiments
    - » Have to anticipate claims that implementation is not correct
- What are your options?
  - Pure implementation – no need to model messy details
  - Simulation – can explore wide parameter ranges



## TCP Over Wireless

- Problem: Wireless links have new loss characteristics
  - Key: Losses are not due to congestion
    - » Sporadic bit-error rates (e.g., temporary blockage)
    - » Losses due to handoffs
  - Sender should not scale back its rate
  - But this is exactly what TCP does in reaction to loss
    - » TCP interprets loss solely as a congestion signal
- Approaches
  - End-to-end: Only change TCP at end-hosts
  - Link-layer: Rexmit at link-layer, hide loss from sender
  - Split-connection: Use two TCP connections, split at base station, to separate congestion from wireless losses

# End-To-End

- End-to-end approaches try to solve the problem by changing TCP at the end points
  - E2E – Reno: fast retransmission, fast recovery (base)
    - » Timeouts when > 1 loss in window, redundant rexmits
  - E2E-NewReno – fast recovery with "partial" acks
    - » Faster recovery, still redundant rexmits
  - E2E-SACK – Sender explicitly notified of missing data
    - » Losses still interpreted as congestion, senders backs off
  - E2E-ELN – Explicit Loss Notification
    - » Receiver tells sender that a loss is not from congestion, sender does not back off (does not shrink congestion window)
  - E2E-ELN-RXMT – ELN w/ faster retransmission
    - » Rexmit on first dup ELN ack



# Link-Layer

- Wireless networks are a shared resource
  - It is essentially a bus, and all devices compete to use it
  - Base station must mediate access to it
  - Many algorithms to do the mediation
    - » We skipped this topic
    - » Ad hoc routing paper describes 802.11 (wireless ethernet)
- Link-layer often tries to recover from losses transparently from higher level layers
  - Key is that link-layer can do retransmissions as well
  - Seen by higher layers only as a longer delay for packet
  - Trade-off between transport rexmits and link rexmits
    - » Don't want link-layer to be perfectly reliable, just reliable enough
    - » Back to the end-to-end principle

# Link-Layer (2)

- Four link-layer approaches
  - LL – Cumulative acks (based on TCP acks), link-layer does local rexmits from base station
    - » LL rexmit timer much shorter than TCP timer, reacts faster
    - » Dup acks still go back to sender, though (invoking cong. avoid.)
  - LL-SACK – Use SACK to only rexmit actual losses
    - » Dup acks still go back to sender
  - LL-TCPAWARE – Suppress dup acks ("snoop", by authors)
    - » Sender does not see wireless losses
    - » Does this violate E2E?
  - LL-OPT – Suppress dup acks, only rexmit actual losses
    - » TCPAWARE + SACK



# Split-Connection

- Split connection approaches use two connections
  - Sender ←→ base station ←→ receiver
  - Note: Overhead of managing two connections
- S←→B connection experiences congestion losses
  - Use normal TCP (Reno)
- B←→R connection experiences wireless losses
  - Can use something better adapted to wireless link
- Two variants
  - SPLIT – Reno TCP over wireless
  - SPLIT-SCAK – SACK TCP over wireless

# Experimental Evaluation

- Methodology
  - Implemented all variants in BSD
  - Experiments using implementation on real wireless network
  - No need to model any of the underlying complexity
    - » c.f. ad hoc routing paper
- Metrics
  - Throughput – total bytes through network
  - Goodput – useful bytes through network
  - WAN tests, too
- Losses artificially induced
  - Reasonable?



# Results

- Paper posed four questions at beginning
  1. What combination of mechanisms results in best performance for each protocol class?
  2. How important is it for link-layer schemes to be TCP-aware?
  3. How useful is SACK?
  4. Is splitting necessary for good performance?
- Also, socket buffer size matters…

# Best Within Approaches

1. What is the best variant in each approach?

- Link-layer → LL-OPT (best overall)
  - Wireless losses ideally handled, sender only sees congestion
  - LL, LL-SACK not as effective – no suppression
  - Best wide-area performance across all approaches
- E2E → E2E-SACK
  - Much lower throughputs than LL (but excellent goodputs)
  - Why would SACK be better than ELN?
- Split → Split-SACK
  - SACK necessary, otherwise performance is terrible



# Remaining Questions…

2. Should LL be TCP aware?
   - Yes: 10-30% better throughput
3. How useful is SACK?
   - Significantly benefits all approaches (message: SACK is good)
4. Is TCP splitting necessary?
   - It gives excellent performance, but LL-TCP-AWARE and LL-OPT are just as good
   - Splitting not necessary
   - Is LL less complex than splitting?

# Discussion

- What does “significant” mean in results?
  - “LL wireless goodput is only 95.5%, significantly less than LL-TCP-AWARE’s wireless goodput of 97.6%”
- Wireless congestion
  - What happens when you do have congestion in wireless?
  - How bad is it to misinterpret a loss due to congestion?
- Constrained setup
  - What about sending the other direction?
- Wireless losses
  - I’m also wondering what type of loss rates we get in these new wireless home lans. Is it substantial?



# Discussion (2)

- Determining source of loss
  - In practice, how can we identify which packets are lost due to errors on all lossy link?

# Ad Hoc Routing In Wireless Networks

- Ad hoc networks are composed of mobile nodes with overlapping cells centered on each node
  - Two nodes are connected by a "link" if their cells overlap
  - No centralized supporting infrastructure (no base stations)
  - Frequent changes in network topology
- Problem: How do proposed routing algorithms perform on these networks?
  - Many routing algorithms proposed
  - Not evaluated well individually
  - Not evaluated on common ground



# Approach

1. Build a detailed simulator (ns) of a wireless network
   - Node mobility
   - Physical layer with radio propagation model
   - Interfaces with power and receiver sensitivity
   - 802.11 MAC protocol for mediating access to media
2. Implement routing algorithms in ns
   - Twice, independently, to validate (rare)
   - Add improvements to algorithms based upon experimentation
     - » Do a better job than original developers
3. Simulate over a wide range of conditions
   - Topologies, movement, sources, etc.
   - 210 scenarios (one advantage of simulation)
4. Write paper using microfont

# Routing Algorithms

- Four routing algorithms analyzed
  1. Destination-Sequenced Distance Vector (DSDV)
  2. Temporally-Ordered Routing Algorithm (TORA)
  3. Dynamic Source Routing (DSR)
  4. Ad Hoc On-Demand Distance Vector (AODV)
- Important phrase in paper:
  - "each based on different assumptions and intuitions"
  - Assumptions and intuitions can sink your ship (e.g., TORA)



# DSDV

- DSDV is distance vector with sequence numbers
  - Forwarding tables with next-hop info broadcast to neighbors
  - Entries contain sequence number in addition to metric
    - » Nodes announce even seq #s for themselves
    - » Nodes announce odd seq #s, infinite metrics for next-hops to destinations when link to next-hop is broken
    - » Destinations react to this with a higher seq #, metrics recomputed at all nodes
    - » In this in-between time, destination is unreachable
      - But loops are prevented
- Variants
  - DSDV – Trigger updates only upon receipt of new metrics
  - DSDV-SQ – Trigger updates upon receipt of new seq #

# TORA

- TORA's goals are to:
  - Discover routes on demand
  - Provide multiple routes to destinations
  - Establish routes quickly
  - Localizing reactions to topology changes
- In TORA
  - Nodes broadcast query packets to discover routes
  - Other nodes that have a route respond with an update
    - » Either destination or intermediate node
  - This update establishes a route from originator to responder
  - When a link goes down, nodes connected send an update with an effectively infinite metric



# TORA (2)

- TORA layered on the Internet MANET Encapsulation protocol (IMEP)
  - Provides reliable, in-order delivery of control messages
    - » Does resends until it gets an ack
  - Notifies upper layer when a link goes down
    - » If it does not eventually receive an ack, it consider the link down
  - Beacons are used to sense link status when no traffic is going across the link
    - » Good or bad?
- IMEP turns out to be Achilles heel of TORA

# DSR

- DSR uses source routing
  - Routes discovered on demand
  - Nodes forward packets according to routes in packets
- Route Discovery
  - Broadcast Route Request, route returned in Route Reply
  - Cache source routes to reduce need for discovery
- Route Maintenance
  - Need to decide when a source route no longer works
  - When a link breaks, send Route Error to senders in cached source routes
  - Sender can use a different source route, or discover another



# AODV

- AODV combines features of DSR and DSDV
  - Route Requests are flooded to discover routes to destinations
  - Nodes create reverse routes back to sender during flooding
  - Route Replies contain hop count and a sequence number
    - » Route Replies sent back to sender via reverse routes
    - » Intermediate nodes create forward routes to destination
    - » Both reverse and forward routes are next-hop (c.f., DSDV)
  - Upon link failure, Unsolicited Route Reply sent upstream to recent senders with infinite metric to break forwarding path
- Variants
  - AODV – use beacon messages to detect failed links
  - AODV-LL – use link-layer feedback to detect failed links

## Experimental Evaluation

- Goal: Determine how routing algorithms react to topology changes while delivering data
- Metrics
  - Packet delivery ratio – packets sent / packets recieved
  - Routing overhead -- # routing packets sent (per-hop)
  - Path optimality – difference between actual path and optimal
- Sources
  - Constant bit rate, not TCP
  - With TCP, could not get comparable transmissions



## Results

- Packet delivery
  - All do well except DSDV-SQ at small pause times
    - » DSDV-SQ fails to converge at high rate of topology changes
  - TORA breaks down with heavy workload (large # sources)
    - » TORA creates short-lived routing loops that interfere with the ability of the network to converge
- Routing overhead
  - DSDV-SQ has overhead independent of rate of change
  - Others depend – DSR smaller than DSDV, others >> larger
  - TORA again breaks down with heavy workload
    - » Sends 100x more packets than other algorithms
    - » Unstable – overhead dramatically increases as load increases

# Results (2)

- Path optimality
  - DSDV-SQ and DSR are close to optimal
  - TORA, AODV-LL have small fraction of paths with 4+ hops
  - How much does this matter?
- Lower speeds (1m/s max speed)
  - All deliver 98.5% of packets at this speed
  - DSR's cache even more effective
  - IMEP a significant source of packets for TORA (beacons)



# Discussion

- Why use a speed of 20m/s instead of 1m/s?
  - What do these speeds correspond to?
  - Is workload too extreme?
- What do the pauses correspond to?
  - Eventually want to ground this in reality
- Why would we not want to use these algorithms within and among ISPs?
  - What makes ad hoc routing different?
- Conclusions
  - What can we do with the drawbacks of those protocols?

# For Next Time...

- Only ST-II/RTSP comparison is required reading