CMPE 257: Wireless and Mobile Networking

Katia Obraczka
Computer Engineering
UCSC Baskin Engineering

Lecture 12

CMPE 257 Spring’15
Student Presentations

- May 21: Sam (propagation models); Anuj (IoT)
- May 26: Larissa (user mobility) and Armando (game theory in MANETs)
- May 28: Daniel (localization and geocasting) and Alan (SDN)
- June 2: Ben (secure MANET routing) and Maziar (ICN)
- June 4: Nokia???
Reading reports

• During student presentation time:
  – Presenter does not need a reading report.
  – Everybody else:
    • Pick 3 papers out of 6 (except survey papers).
Reliable Point2Point Transport Layer: Outline

- TCP/IP basics.
- Impact of transmission errors on TCP performance.
- Approaches to improve TCP performance on wireless networks.
  - Classification.
- TCP on cellular.
  - TCP on MANETs.
TCP in Mobile Ad Hoc Networks
Improving TCP under OOO Delivery [Wang02]
Out-of-Order Packet Delivery

- Route changes may result in out-of-order (OOO) delivery.
- Significantly OOO delivery confuses TCP, triggering congestion control.
- Potential solutions:
  - Avoid OOO delivery by ordering packets before delivering to TCP layer.
  - Turn off fast retransmit.
    - Can result in poor performance in presence of congestion.
TCP DOOR

• Strictly E2E approach.
• Detect and respond to out-of-order (OOO) packets.
  – Differentiate between OOO and congestion losses.
• OOO delivery caused by:
  – Retransmissions.
  – Route changes.
Detecting OOO

- OOO delivery can happen in either direction.
- Sender detects OOO (duplicate) ACKs.
- Receiver detects OOO data packets.
OOO ACKs

• At sender.
• Detecting OOO delivery of non-duplicate ACKs is simple.
  – Sequence number of packet being ACKed: monotonically increasing.
• For DUPACKs, add 1-byte to count DUPACKs.
  – ADSN: ACK duplication sequence number.
  – TCP header option.
  – Each DUPACK carries different ADSN.
OOO Data Packets

• At receiver.
• Why comparing sequence numbers doesn’t work?
  – Retransmissions carry the same sequence number as the original transmission.
  – Higher sequence #’s can arrive earlier.
• Use extra sequence number: incremented with every data packet, including retransmissions.
  – 2-byte TCP packet sequence number (TPSN) as TCP option.
  – Or timestamp.
• Sender needs to be notified.
OOO Response

• At sender.
  – If receiver detects OOO event, it should notify the sender.

• 2 types of response:
  – Temporarily disable congestion control for fixed time interval T1.
  – If in congestion avoidance mode in the last T2 time interval, go back to prior state.
Evaluation

• Simulation environment:
  – ns-2 + CMU extensions for MANETs.
  – Mobility: random way-point.
  – DSR route caching.
  – Workload: single TCP between fixed S and R with and without congestion.
Results

• Significant goodput improvement (~50%) when 2 response mechanisms used.
• Sender versus receiver detection.
  – Seem to perform the same.
  – Correlation between OOO ACKs and data.
• Response mechanisms.
  – When both are in place, better performance results.
DSR Caching

• With DSR caching enabled, lower performance improvements.
  – Why?
(Reliable) End-to-End Protocols

• Reliable point-to-point communication.
• Reliable multipoint communication.
Reliable Multipoint Communication in MANETs
Motivation

• Group communication services as important application class for key MANET scenarios: special operations, emergency response, etc.
  – E.g., teleconferencing and data dissemination.

• Need efficient way of delivering many-to-many data reliably.
Challenges

• Achieve reliability in the presence of constant mobility, route changes, transmission errors over multi-hop wireless routes.
  – Different reliability semantics.

• MANETs are very sensitive to load and congestion.
  – Contention-based MACs.

• Not much had been done…
Reliable Broadcast
[Pagani1997]

• Special case: all nodes are receivers.
• Reliable broadcast: all nodes deliver same set of messages to application.
  – “Exactly-once” semantics.
• Assumes underlying clustering algorithm.
Primitives

• `rbcast (msg, group-id)`
  – Caller blocks until message received by clusterhead.

• `deliver(msg)`
  – Notification of message reception by all other nodes.
Protocol Entities

- End node
- Gateway
- Clusterhead
Protocol Entities

- Gateways and end nodes are stateless.
- Protocol state kept at clusterheads
Protocol

- Sender unicasts message to its clusterhead.
- Clusterhead broadcasts message within cluster and waits for ACKs.
- Nodes reply with ACK.
- Gateways forward message to directly connected clusters (through clusterheads).
- Gateway delays its ACK until it gets ACK from corresponding clusterheads.
Protocol (Cont’d)

• If same message received over multiple paths: clusterhead selects one; piggybacks the sender id in the message and broadcasts.
• Non-selected gateway receives the broadcast and records it’s the leaf for that message.
• Prevents loops, allows multi-path, reduces duplicates.
• Reverse path is recorded: ACKs flow back.
Figure 2: (a) The broadcast protocol in the case of static topology. (b) The corresponding $FT$. 
Failures and Mobility

- Timeouts and retransmissions.
- Recovery by clusterheads & gateways.
Summary

- 2 phases:
  - Diffusion: message diffused from source to destinations.
  - Gathering: source collects ACKs.
- On-demand forwarding tree rooted at source.
- If “virtual” links break, flooding is used.
- Clusterheads buffer messages until ACK comes back.
- Reliability guaranteed as long as “liveness” property holds.
  - Topology is stable enough.
Issues

• Larger networks?
  – High delays.
  – Target smaller groups (10->100???).

• What happens when clusterheads and gateways fail/move?
  – Satisfying “liveness” is tricky.

• Heavy duty protocol.
  – State at nodes.
  – ACKs for reliability.
Anonymous Gossip
[Chandra2001]

• Approach:
  – Use of underlying “best-effort” multicast routing mechanism.
  – Repair losses through gossip-based propagation.

• Gossip propagation is well-known!
  – Examples?

• Gossip is quite robust!
Gossip Round

• Node A randomly chooses node B.
• A and B exchange information on messages they have and don’t have.
• A and B exchange missing messages.
Anonymity

• No need for membership information.
• \texttt{gossip-message} and \texttt{gossip-reply}.
  – Node selects random neighbor and sends \texttt{gossip-message}.
  – If node is not group member, selects neighbor and forwards; if member, decides to accept or forward.
  – If accepts, unicasts \texttt{gossip-reply} back.
Locality

• Choosing closer-by or farther members.
  – Why is this an issue?
  – Choose near members with higher probability.
  – Need to keep extra state: nearest-member.
    • Distance to nearest group member from this node.
    • Extra overhead to maintain this information.

• Dependence on the underlying routing protocol!
Cached Gossip

- Gossip with known members.
- *member-cache* keeps information on known members.
  - Updated when messages are received (data, *gossip-reply*, RREQ, etc.).
- Node uses AG with probability $p_{anon}$; otherwise cached gossip.
Performance Evaluation

- Simulations using GloMoSim.
- M-AODV.
- Single source.
- Random way-point.
- Parameters: range, number of nodes, and maximum node speed.
Results

- Improved packet delivery ratio.
- Overhead?
- Delay?
- Comparison with flooding?
CALM [Tang2002]

• Like AG, e2e approach.
• Initial study comparing SRM to UDP showed SRM performs badly in MANETs.
• Why?
  – SRM is heavy duty.
  – No congestion control!
Decoupling Congestion Control and Reliability

• CALM uses “rate-based” CC.
  – Data is sent at application rate.
  – If congestion (NACK), sender clocks sending rate based on receivers experiencing congestion.

• Congested receivers kept in receiver-list.
CALM

• Source multicasts next data packet to selected receiver in receiver-list.
• Source explicitly requests problematic receiver to ACK.
• If ACK received before time-out, receiver removed from receiver-list.
• When receiver-list empty, source reverts back to nominal application sending rate.
• Feedback is unicast to the source.
  – Generated after N consecutive packets are missing.
Evaluation

- Simulations using QualNet.
- Comparison between CALM, SRM, and (multicast) UDP.
- ODMRP.
Results

• CALM outperforms SRM.
• UDP performs surprisingly well, except under high traffic loads.
  – Proves need for congestion control.
• SRM’s main problem is extra load caused by its control packets.
• Congestion control helps but still need reliability.
RALM [Tang03]

- Reliability+congestion control.
RALM Overview

• Source initially sends at application’s nominal rate.
• If NACK, source enters “loss recovery”.
  – Receivers experiencing losses added to “receiver list”.
  – Source selects receiver from “receiver list” to reliably transmit: “feedback receiver”.
  – Source **multicasts** lost packet(s) to feedback receiver.
    • Feedback receiver **unicasts** ACKs/NACKs.
    • Only feedback receiver sends feedback.
RALM Overview (cont’d…)

• During loss recovery:
  – Source retransmits lost packets one at a time: stop-and-wait. Why?
  – When receiver list is empty, source goes back to application sending rate.
RALM

Fig. 2. RALM state transition diagram at the source.
Results

• From paper...
ReACT [Rajendran04]

- Source-based + local recovery.
- Receiver-based loss differentiation.
  - Distinguish between “global congestion” and local losses.
  - Use cross-layer information.
  - Multicast receiver samples its MAC queue.
  - Intermediate nodes also report if their MAC queue grows above certain threshold.
- Local recovery: uses non-congested nodes.
ReACT: Source-based control

- Rate-based CC.
- Initial rate establishment and congestion control modes.
- At initial rate establishment, source decides the initial sending rate.
  - Probes directly connected members.
  - Uses inverse of longest measured RTT to set source’s initial rate.
  - Rate periodically updated based on feedback from directly connected receivers.
ReACT: Source-based control (cont’d…)

• Congestion control mode:
  – Similar to RALM.
ReACT: Receiver-based recovery

• Detect losses that can be recovered locally.
• Receivers need to know about other local group members.
• For local recovery, node chooses non-congested member with highest receive rate, lowest hop count and latest timestamp.
  – Source route to recovery node.
  – Route, hop count and congestion information are updated as packet is forwarded.
ReACT: Receiver-based recovery

- NACK with missing sequence numbers sent to recovery node if losses are local.
- Otherwise, NACK is forwarded to source.
- Losses are global if all paths to recovery nodes are congested or node’s local queue is getting full.
Evaluation

• Simulations.
• Comparison against RALM.
• Metrics: reliability, (reliable) throughput, overhead.
• Effect of congestion and mobility.
Results

• ReACT achieves better reliability and goodput with lower overhead.
  – Local recovery prevents source from reducing rate unnecessarily.
  – Local recovery also lowers overhead.