

CMPE 257: Wireless and Mobile Networking

Katia Obraczka
Computer Engineering
UCSC Baskin Engineering

Lecture 11

CMPE 257 Winter'11



Inter-Networking Research Group

Student Presentation: Logistics

- We will source from San Jose.
 - Send me your presentations ahead of time.

Today

- E2E protocols (cont'd).

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 infrastructure-based networks (cont'd).
- TCP on MANETs.

Strict E2E Schemes

Receiver-Based Scheme

[Biaz98Asset]

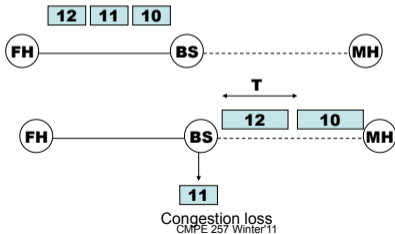
- MH is TCP receiver.
- Receiver uses heuristics to “guess” cause of packet loss.
- When receiver believes that packet loss is due to errors, it sends notification to sender.
- TCP sender, on receiving notification, retransmits lost packet, without reducing congestion window.

Heuristics

- Receiver uses inter-arrival time between consecutively received packets to guess cause of packet loss.
- On determining a packet loss as being due to errors, the receiver may:
 - Tag corresponding dupacks with an ELN bit, or
 - Send an explicit notification to sender.

Receiver-Based Scheme

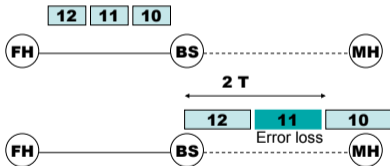
- Packet loss due to congestion:



Congestion loss
CMPE 257 Winter'11

Receiver-Based Scheme

- Packet loss due to transmission error:



Sender-Based Discrimination Scheme

Sender-Based Discrimination

[Biaz98ic3n,Biaz99techrep]

- Sender can attempt to determine cause of a packet loss.
- If packet loss determined to be due to errors, do not reduce congestion window.
- Sender can only use statistics based on round-trip times, window sizes, and loss pattern.
 - Unless network provides more information (example: explicit loss notification)

Heuristics for Congestion Avoidance

- Define condition C as a function of congestion window size and observed RTTs.
- Condition C evaluated for new RTT.
- If (C == True), reduce congestion window.

Heuristics for Congestion Avoidance: Some proposals

- TCP Vegas [Brakmo94]

expected throughput $ET = W(i) / RTT_{min}$

actual throughput $AT = W(i) / RTT(i)$

Condition C = ($ET - AT > \beta$)

Sender-Based Heuristics

- Record latest value evaluated for condition C.
- When a packet loss is detected:
 - If last evaluation of C is TRUE, assume packet loss due to **congestion**.
 - Else assume packet loss due to transmission errors.
- If packet loss determined to be due to errors, do not reduce congestion window

Sender-Based Heuristics: Disadvantage

- Does **not** work quite well enough!!

Reason

- Not much correlation between observed short-term statistics, and onset of congestion.

Sender-Based Heuristics:

Advantages

- Only sender needs to be modified.

Needs further investigation to develop better heuristics.

- Investigate longer-term heuristics.

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

Issues

- Route changes due to mobility.
 - Frequent route changes may cause OOO delivery.
- Wireless transmission errors.
 - Problem compounded due to multiple hops.
- MAC
 - MAC protocol can impact TCP performance.

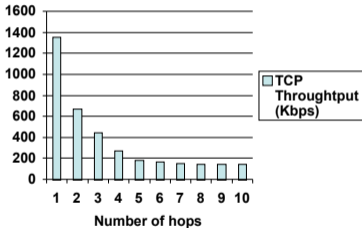
Throughput over Multi-Hop Wireless Paths [Gerla99]

- When contention-based MAC protocol is used, **connections over multiple hops are at a disadvantage** compared to shorter connections.
 - They have to contend for wireless access at each hop.
 - Delay or drop probability increases with number of hops.

Analysis of TCP Performance over MANETs [Holland99]

- Impact of mobility.
- Simulation study.
- Performance metric: throughput.
 - Baseline: ideal (expected) throughput.
 - Upper bound.
 - Static network.

Throughput versus Hops



TCP throughput over 2 Mbps 802.11 MAC,
fixed, linear MANET.

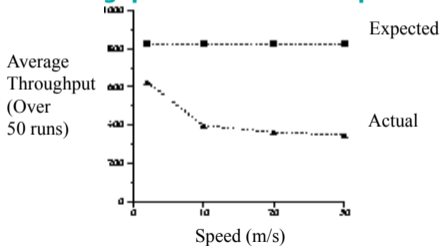
Expected Throughput

$$\text{expected_throughput} = \frac{\sum_{i=1}^{\infty} t_i * T_i}{\sum_{i=1}^{\infty} t_i}$$

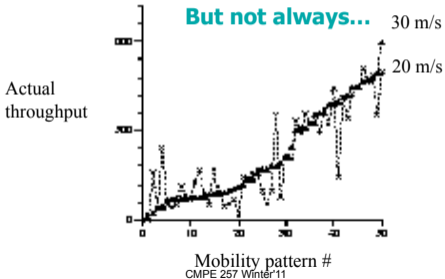
- T_i is measured throughput for i hops using static linear chain topology.
- t_i time duration of TCP connection containing i hops.

Throughput **versus** speed

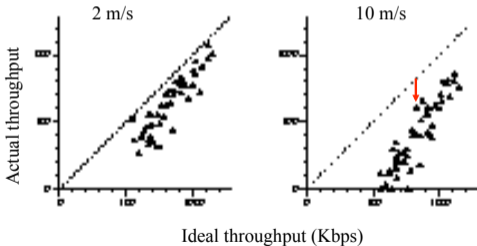
Throughput decreases with speed...



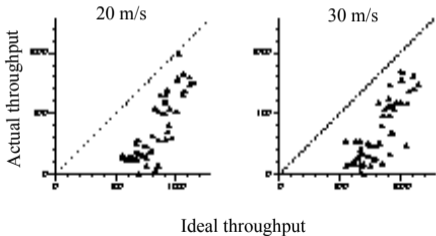
Throughput versus Speed



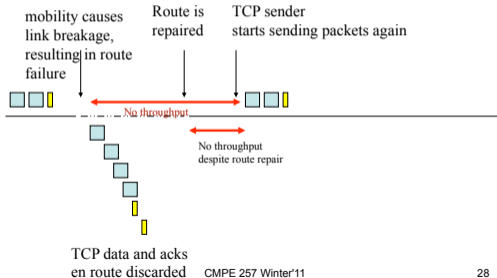
Impact of Mobility TCP Throughput



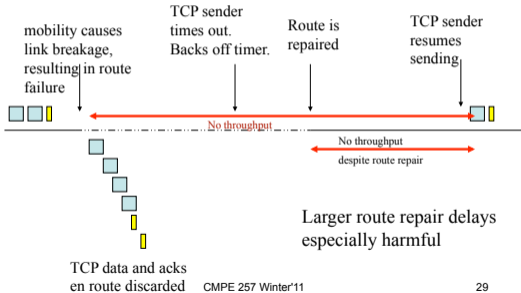
Impact of Mobility



Why Throughput Degrades

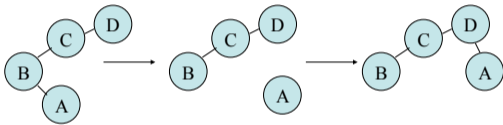


Why Throughput Degrades?



Why Throughput Improves?

Low Speed Scenario



1.5 second route failure

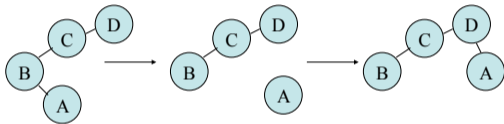
Route from A to D is broken for ~1.5 second.

When TCP sender times out after 1 second, route still broken.

TCP times out after another 2 seconds, and **only then resumes**.

Why Throughput Improves?

Higher Speed Scenario



0.75 second route failure

Route from A to D is broken for ~ 0.75 second.

Before TCP sender times (after 1 second), route is repaired.

Why Throughput Improves?

General Principle

- TCP timeout interval somewhat independent of speed.
- Network state at higher speed, when timeout occurs, may be more favorable than at lower speed.
- Network state:
 - Link/route status.
 - Route caches.
 - Congestion.

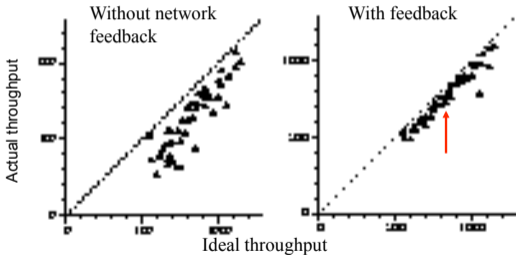
How to Improve Throughput

- Network feedback.
- Inform TCP of route failure explicitly.
- Let TCP know when route is repaired.
 - Probing.
 - Explicit notification.
- Reduce repeated TCP timeouts and backoff.

ELFN

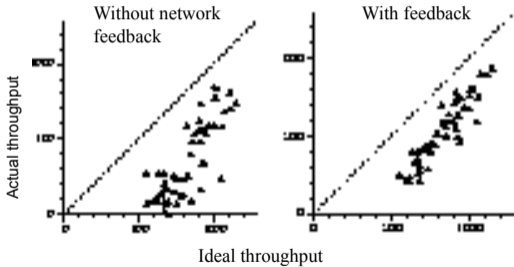
- Explicit Link Failure Notification.
- Piggyback notification onto DSR's route failure message to sender.
- TCP responds by disabling congestion control until route is fixed.
 - Disable retransmission timers.
 - When ACK is received, TCP restores state and resumes normal operation.

Performance Improvement



2 m/s speed
CMPE 257 Winter 11

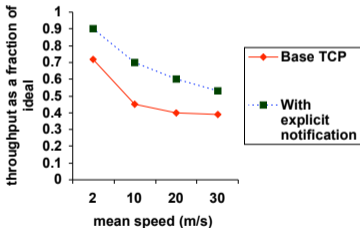
Performance Improvement



30 m/s speed

CMPE 257 Winter 11

Performance with Explicit Notification



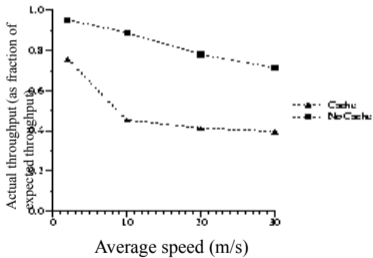
Issues: Network Feedback

- Network knows best (why packets are lost).
- + Network feedback beneficial.
- Need to modify transport & network layer to receive/send feedback
- Need mechanisms for information exchange between layers.

Impact of Caching

- Route caching has been suggested as a mechanism to reduce route discovery overhead (e.g., DSR).
- Each node may cache one or more routes to given destination.
- When route from S to D detected as broken, node S may:
 - Use another cached route from local cache, or
 - Obtain a new route using cached route at another node.

To Cache or Not to Cache



Why Performance Degrades With Caching

- When a route is broken, route discovery returns cached route from local cache or from nearby node.
- Cached routes may also be broken.



To Cache or Not to Cache

- Caching can result in **faster** route “repair”.
- But, faster does not necessarily mean **correct**.
- If incorrect repairs occur often enough, caching performs poorly.
- Need mechanisms for determining when cached routes are stale.

Caching and TCP Performance

- Caching can reduce overhead of route discovery even if cache accuracy is not very high.
- But if cache accuracy is not high enough, gains in routing overhead may be offset by loss of TCP performance due to multiple timeouts.

Window Size After Route Repair

- When route breaks: may be too **optimistic** or may be too **conservative**.
- **Better be conservative** than overly optimistic
 - Reset window to small value after route repair.
 - TCP needs to be aware of route repair (Route Failure and Route Re-establishment Notifications).
 - Impact low on paths with small delay-bw product.

RTO After Route Repair

- If new route longer, RTO may be too small, leading to timeouts.
- New RTO = function of old RTO, old route length, and new route length.
 - Example: $\text{new RTO} = \text{old RTO} * \text{new route length} / \text{old route length}$
 - Not evaluated yet.

TCP Over Different Routing Protocols [Dyer2001]

- Impact of routing algorithm on TCP performance.
 - Metrics: connect time, throughput and overhead.
- On-demand routing.
 - AODV and DSR.
 - ADV: adaptive on-demand with proactive updates.
- Sender-based heuristic to improve TCP's performance.

Fixed-RTO

- TCP does not exponentially backoff the RTO.
- Uses **sender-based** heuristic to distinguish between congestion and “route failure” losses.
 - Route failure assumed if 2 consecutive timeouts.
 - Unack'd packet retransmitted.
 - No RTO backoff in the second (and +) timeout.
 - RTO remains fixed until retransmission is ack'd.

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

- 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

- Sequence number of packet being ACKed: monotonically increasing.
 - Why? ACKs are not re-transmitted.
- 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: higher sequence #'s can arrive earlier.
 - Out-of-sequence event.
- 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.
- 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.
 - Mobility: random way-point.
 - 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.
 - Both in place show better performance.

DSR Caching

- With DSR caching enabled, lower performance improvements.
- Claim TCP performance was better than when caching was off.
 - Why?