

TCP over Wireless Networks

Raj Jain
Washington University
Saint Louis, MO 63131
Jain@cse.wustl.edu

These slides are available on-line at:

<http://www.cse.wustl.edu/~jain/cse574-06/>



- ❑ TCP: Key Features
- ❑ TCP Congestion Mechanisms
- ❑ Our Initial Research on TCP Congestion
- ❑ TCP Over Wireless: Issues and Solutions
- ❑ TCP over Satellite
- ❑ Our research on TCP over Satellite and Wireless

TCP: Key Features

- 1. Stream-Oriented Transmission:** Multiple application packets may be send in one TCP “**Segment.**”
Maximum Segment Size (**MSS**).
All acks are byte numbers. Segment # used in all discussions.
- 2. Reliable Delivery:** Segments are buffered at the source until acked. Retransmitted if not acked.
- 3. In-Order Delivery:** Destination delivers segments to application only when all previous segments received.
- 4. End-to-End Semantics:** Ack \Rightarrow Data received at destination
- 5. Congestion Control:**
Increases load slowly from a low initial start.
Reduces load if network congested (based on segment timeout, duplicate acks)
- 6. Congestion Avoidance:** Explicit Congestion Notification (ECN) bits in TCP/IP headers based on DECbit research

TCP Flow Control

- ❑ **Cumulative Acks:** Acks all bytes up to the ack
- ❑ **Piggybacked Acks:** Acks are sent in the reverse packets if possible.
- ❑ **Delayed Acks:** Ack delayed in case another segment is received or segment needs to be sent. Typically 200 ms
- ❑ **Duplicate Acks:** If an out of order packet is received, the previous ack is resent. Duplicate acks are not delayed.
- ❑ **Window Flow Control:**
 - $\text{Throughput} = \text{Window} / \text{Round Trip Time}$
 - ❑ Ideal Window Size = Round Trip Time * Link Capacity
= Delay-bandwidth product
 - ❑ TCP sets retransmission timer for only one packet. If the ack is not received and the timer expires, the packet is assumed lost.

Timeout Calculations

- **Old Method:** Using only the mean of measured round trip

$$error = sampleRTT - mean$$

$$mean += a * error \quad /* 0 \leq a \leq 1 */$$

$$timeout = d * mean \quad /* d \geq 1 */$$

RFC 793 suggested 0.1 to 0.2 for a, and 1.3 to 2.0 for d.

- **New Method:** Using mean and standard deviation

$$error = sampleRTT - mean$$

$$mean += a * error \quad /* 0 \leq a \leq 1 */$$

$$sample_dev = |error|$$

$$dev_error = sample_dev - mean_dev$$

$$mean_dev += b * (sample_dev - mean_dev) \quad /* 0 \leq b \leq 1 */$$

$$timeout = mean + c * mean_dev \quad /* c \geq 0 */$$

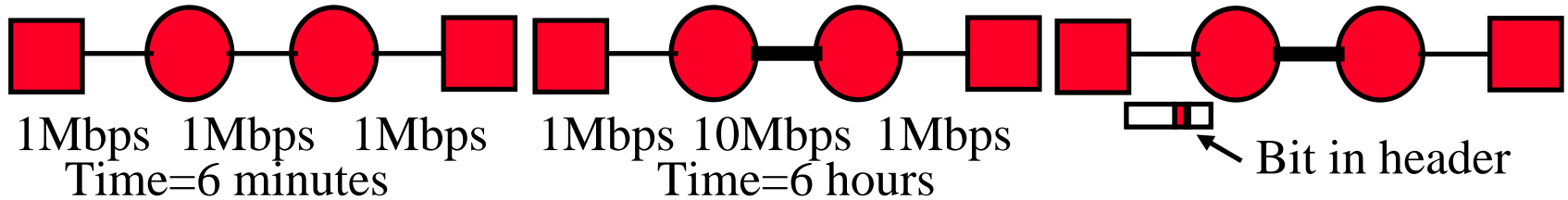
The usual values for the constants are $a = 1/8$, $b = 1/4$, $c = 4$.

- RTT is measured in multiples of a “tick.” 1 tick = 500 ms usually. RTO is at least 2 ticks. Double RTO on repeated timeouts.

TCP Congestion Mechanisms

- ❑ Slow Start
- ❑ Fast retransmit and recovery
- ❑ New Reno
- ❑ Selective Acknowledgement
- ❑ Explicit Congestion Notification

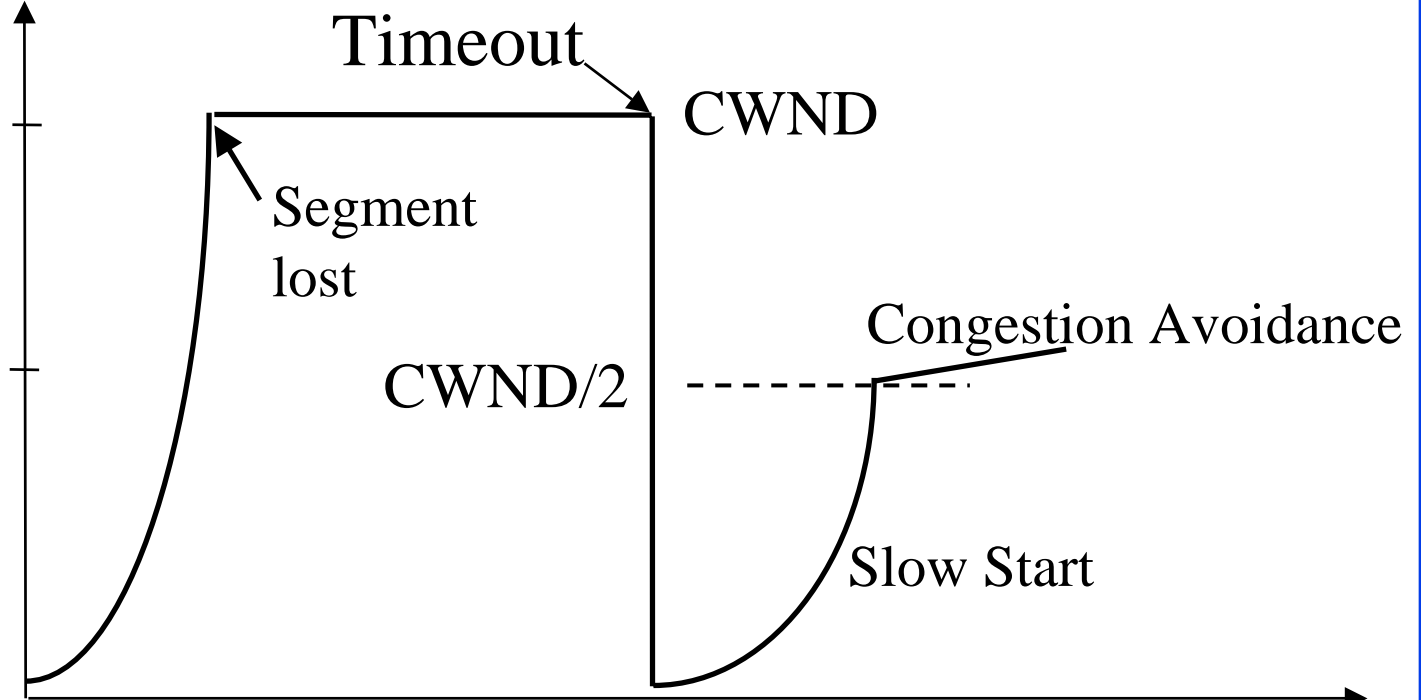
Our Research on TCP Congestion



- ❑ Early 1980s Digital Equipment Corporation (DEC) introduced Ethernet products
- ❑ Noticed that throughput goes down with a higher-speed link in middle (because no congestion mechanisms in TCP)
- ❑ Results:
 1. Timeout \Rightarrow Congestion
 \Rightarrow Reduce the TCP window to one on a timeout [Jain 1986]
 2. Routers should set a bit when congested (DECbit).
[Jain, Ramakrishnan, Chiu 1988]
 3. Introduced the term “Congestion Avoidance”
 4. Additive increase and multiplicative decrease (AIMD principle)
[Chiu and Jain 1989]
- ❑ There were presented to IETF in 1986.
 \Rightarrow Slow-start based on Timeout and AIMD [Van Jacobson 1988]

Slow Start

Congestion Window



TIME

Slow Start Wait for Timeout Slow Start Congestion Avoidance

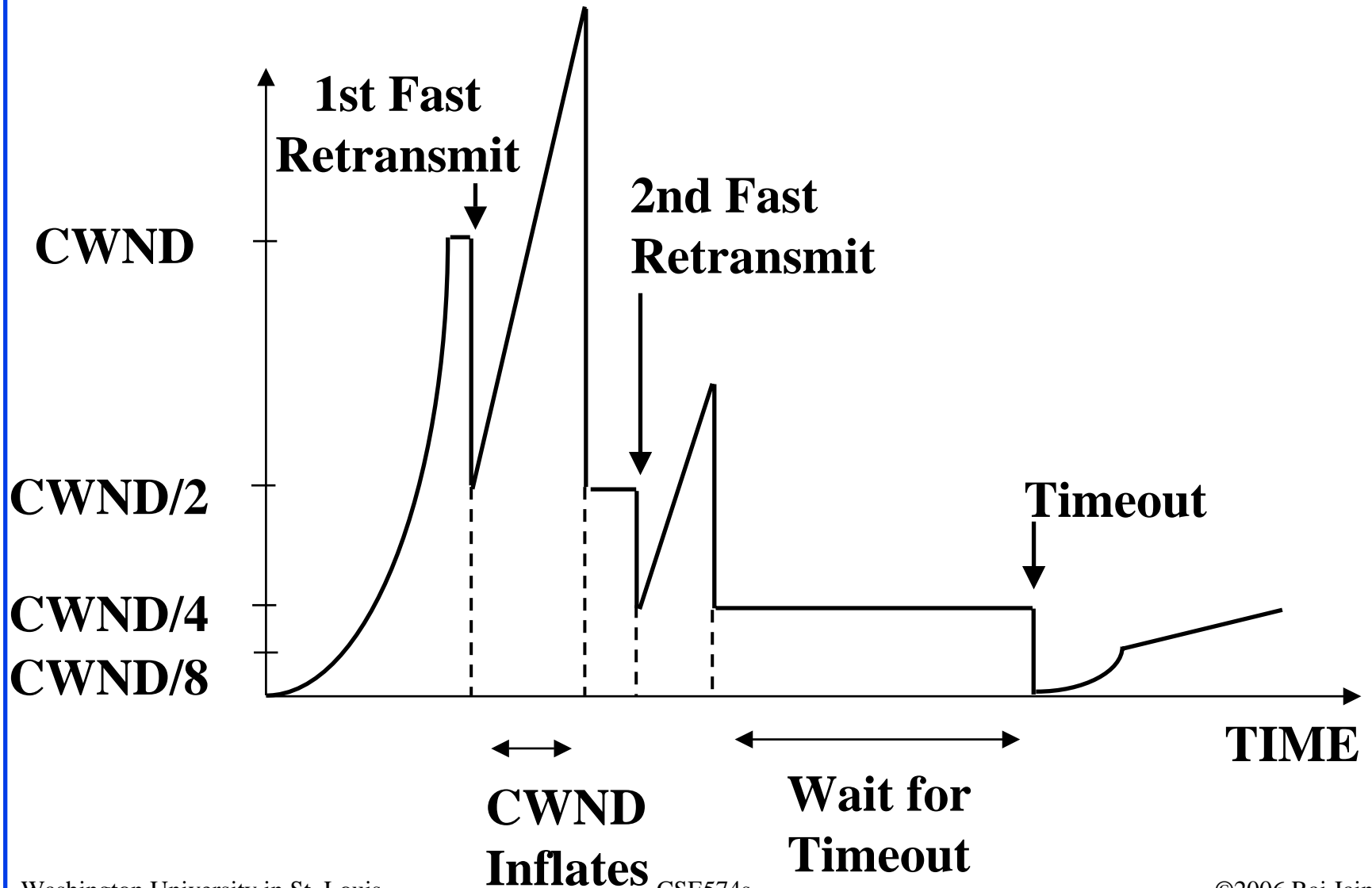
Slow Start (Cont)

- ❑ Receiver sends “Receive window” (for flow control)
- ❑ Sender maintains a **Congestion Window** (CWND)
CWND $W \leq$ Receiver Window
- ❑ Set “Slow Start Threshold”
SSThresh = 64 kB initially
- ❑ Start with a CWND W of 1
- ❑ Increase W by 1 after every ack until SSThresh
 \Rightarrow Exponential increase (**Slow Start**. W doubles every RTT)
- ❑ Increase W by $1/W$ after every Acks (W increases by per RTT)
 \Rightarrow Linear increase (**Congestion Avoidance**)
- ❑ On a timeout, Set SSThresh to half the current window and set window to 1.
 $SSThresh \leftarrow \text{Max}\{2, 0.5W\}, W \leftarrow 1$

Fast Retransmit and Recovery (FRR)

- ❑ Also known as TCP-Reno
- ❑ Ideas:
 - Don't have to wait for timeout on a loss
 - Don't reduce to one on single loss
 - Duplicate acks \Rightarrow Loss
- ❑ On three duplicate acks:
 - Retransmit the lost segment (Fast Retransmit)
 - Set $SSThresh$ to $\text{Max}\{2, 0.5 \times CWND\}$
 - Reduce $CWND$ to $0.5 \times CWND + \# \text{ of dupacks}$
 - New ack $\Rightarrow CWND > SSThresh \Rightarrow$ Linear increase
 - Duplicate ack \Rightarrow inflate $CWND$ by 1. Send a pkt if allowed
- ❑ **Advantage:** Recovers from loss without a timeout
- ❑ **Problem:** Cannot recover from bursty (3+) losses.
Dupacks are also generated if pkts out-of-order (no loss).

FRR (Cont)



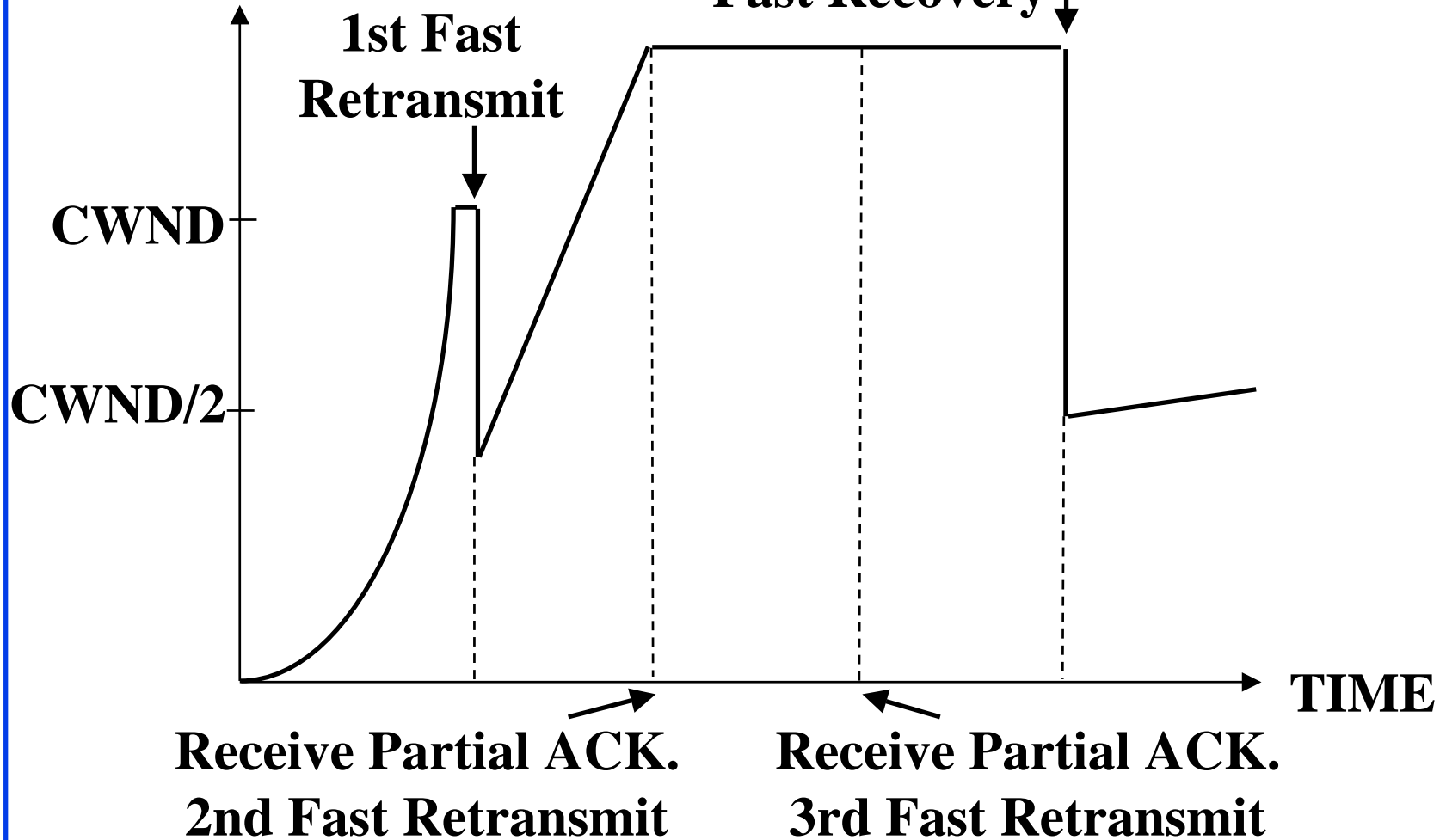
TCP New Reno

- ❑ Janey Hoe's MS Thesis from MIT
Published in SIGCOMM'96
- ❑ Solution: Determine the end-of a burst loss
Remember the highest segment sent (RECOVER)
 $\text{Ack} < \text{RECOVER} \Rightarrow \text{Partial Ack}$
 $\text{Ack} \geq \text{RECOVER} \Rightarrow \text{New Ack}$
- ❑ New Ack \Rightarrow Linear increase from $0.5 \times \text{CWND}$
- ❑ Partial Ack \Rightarrow Retransmit next packet,
let window *inflate*
- ❑ Recovers from N losses in N round trips

New Reno (Cont)

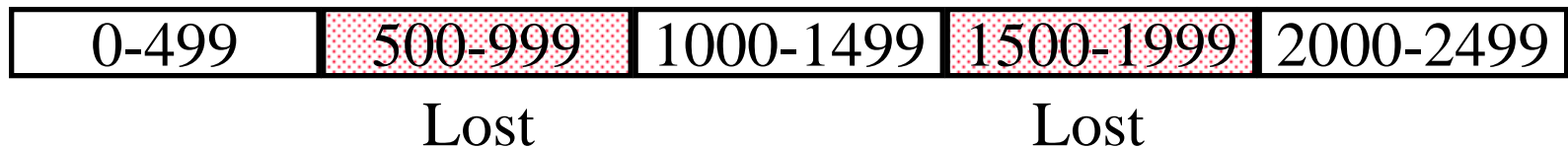
Receive New ACK.

Fast Recovery ↓



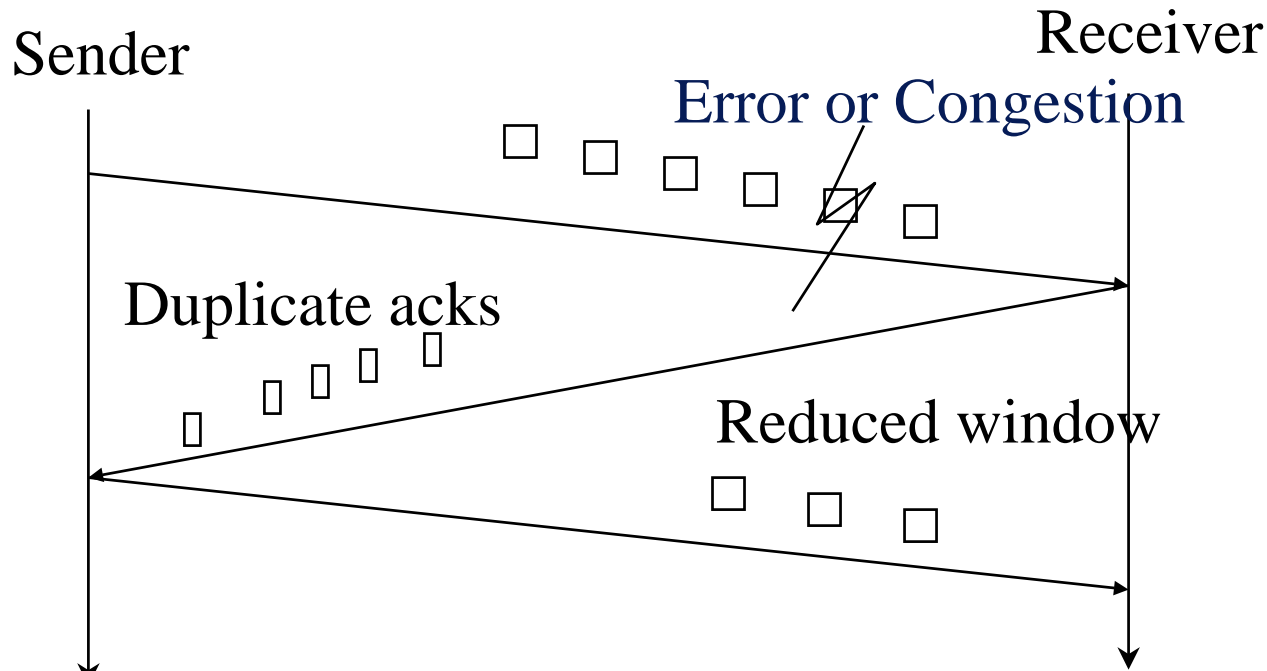
Selective Ack

- ❑ RFC 2018, October 1996
- ❑ Receivers can indicate missing segments
- ❑ Example:
Using Bytes: Ack 500, SACK 1000-1500, 2000-2500
⇒ Rcvd segment 1, lost 2, rcvd 3, lost 4, rcvd 5
- ❑ On a timeout, ignore all SACK info
- ❑ SACK negotiated at connection setup
- ❑ Used on all duplicate acks



Problems of Current TCP

- ❑ TCP cannot distinguish wireless errors from congestion.
- ❑ Frequent errors \Rightarrow Frequent window reductions
 \Rightarrow Low throughput
- ❑ On CDMA, Overload \Rightarrow Errors. Otherwise no relationship.

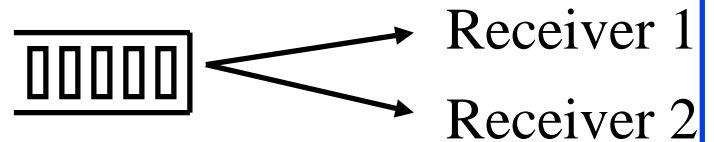


TCP Over Wireless

- ❑ Link Layer Mechanisms
- ❑ Split TCP Solutions
- ❑ TCP Aware Link Layer Protocols
- ❑ Explicit Notification Schemes
- ❑ TCP Over Satellite
- ❑ Our Results for Satellite and Wireless Networks

Link Layer Mechanisms

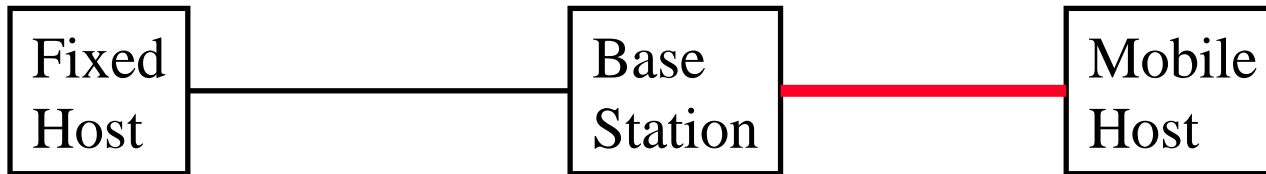
- ❑ Forward Error Correction (FEC):
 - Reduces loss due to errors.
 - Reduced link throughput even if no errors.
- ❑ Automatic Repeat Request (ARQ):
 - Link layer retransmission and acknowledgement
 - No reduction in throughput if no errors
 - Reduced throughput and increased delay at link layer
 - May cause congestion
 - May increase variance of RTT \Rightarrow Increased RTO
 - May cause head-of-line blocking
- ❑ Adaptive Link layer strategies:
 - Dynamically vary FEC code, retransmission limit, frame size



Split TCP Solutions

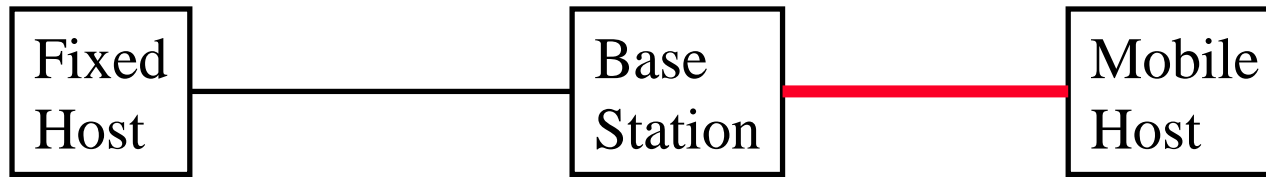
- ❑ Indirect TCP
- ❑ Selective Repeat Protocol (SRP)
- ❑ Mobile TCP
- ❑ Mobile-End Transport Protocol

Indirect TCP



- ❑ Two TCP connections:
 - Fixed host to Base
 - Base to Mobile
- ❑ Independent flow control on two connections
- ❑ Packets buffered in the base
- ❑ Ack at sender \neq MH has received
 - Violates TCP's end-to-end semantics
 - BS retains hard state. BS failure \Rightarrow loss of data
 - On handoff, stored packets must be sent to new BS
 - Does not work if connection not bi-directional. E.g., satellites

Selective Repeat Protocol (SRP)



- ❑ Two connections: Similar to Indirect TCP
- ❑ FH to BS: Standard TCP
- ❑ BS to MH: Selective repeat protocol on UDP
- ❑ Reference: Yavatkar94

Mobile TCP

- ❑ Asymmetric split connection
- ❑ Simpler protocol at mobile host
- ❑ Mobile does error detection only
- ❑ Base does error correction and error detection
- ❑ Header compression on wireless hop
- ❑ On/off flow control on wireless hop
- ❑ Ref: Haas97

Mobile-End Transport Protocol

- ❑ TCP runs only between fixed host and BS
- ❑ BS guarantees reliable ordered delivery to mobile
- ❑ Ref: Wang98

TCP Aware Link Layer Protocols

- ❑ Snoop Protocol
- ❑ WTCP
- ❑ Delayed DupAcks Protocol
- ❑ SCPS-TP

Snoop Protocol

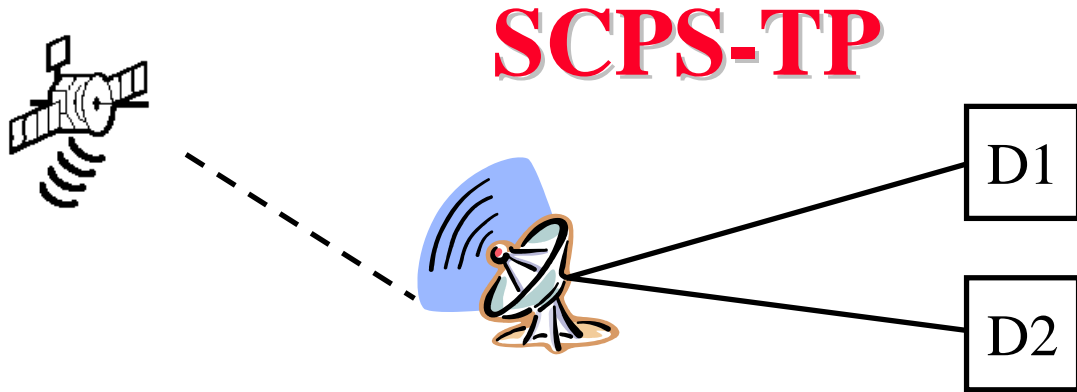
- ❑ Split connection and link level retransmission
- ❑ Base monitors returning acks. Retransmits on duplicate acks and drops the duplicate ack
- ❑ Advantages: Only soft state at BS. Only BS modified. No changes to FH or MH.
- ❑ If wireless link delay is less than 4 packets, 3 duplicate acks will not happen and a simple link-level retransmission without dropping duplicate ack will also work.
- ❑ Disadvantages: Does not work with encrypted packets
- ❑ Does not work on asymmetric paths
- ❑ Ref: Balakrishnan95

WTCP

- ❑ Similar to Snoop
- ❑ Snoop can cause increased RTT
- ❑ WTCP corrects RTT by modifying the timestamp in returning acks
- ❑ Disadvantages:
 - Useful only if retransmission times are large (>1 tick)
 - Does not work on shared LANs, where overload \Rightarrow Increased delay
- ❑ Ref: Ratnam98

Delayed DupAcks Protocol

- ❑ Similar to Snoop. But no duplicate ack dropping at BS
- ❑ Link layer retransmission on wireless hop
- ❑ Third duplicate acks delayed at MH. BS does not need to look into TCP headers.
- ❑ Out-of-order packet delivery from link layer to TCP allowed at MH to avoid head-of-line blocking at MH
- ❑ Advantage: BS is not TCP aware. Can be used even if headers are encrypted.
- ❑ Disadvantages: Congestion losses are recovered later since dupacks delayed.
- ❑ Ref: Mehta98, Vaidya99



- ❑ Space Communications Transport Protocol (SCPS-TP)
- ❑ Used in satellite communications
- ❑ Ground stations monitors packets with failed checksum and sends corruption experienced messages to destinations of recent error-free packets
- ❑ Ground stations can detect outage of incoming link and assume outage of outgoing link also.
- ❑ Destinations ack with “Corruption experienced” bit
- ❑ After receiving an ack with “Corruption Experienced” bit, sender does not back off on timeout or duplicate acks until it receives an ack without that bit.

Explicit Notification Schemes

- ❑ Explicit Loss Notification
- ❑ Explicit Loss Notification 2
- ❑ Explicit Bad State Notification
- ❑ Partial Ack Protocols

Explicit Loss Notification



- ❑ Works with Mobile host sources
First link on the path is wireless
- ❑ BS keeps track of missing packets from mobile
- ❑ When DupAcks is received, BS sets “ELN” bit in the DupAcks
- ❑ When mobile receives the DupAcks with ELN bit, it does not back off. Simply retransmits.

- ❑ Reference: Balakrishnan98

Explicit Loss Notification 2

- ❑ Similar to ELN. Works when receiver is mobile.
- ❑ Caches TCP sequence numbers at the base (as in snoop) but does not cache data packets
- ❑ Dupacks are tagged with ELN bit if sequence number of lost packet is cached at the base
- ❑ Ref: Biaz99

Explicit Bad State Notification

- ❑ Works when Mobile is the receiver
- ❑ Link layer retransmission on the wireless link
- ❑ If base cannot deliver the packet to Mobile, it sends a “explicit bad state notification” (EBSN) message to sender
- ❑ Ref: Bakshi97

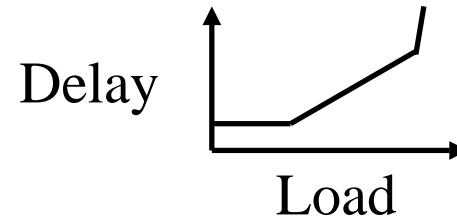
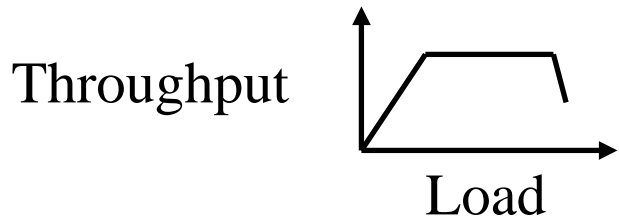
Partial Ack Protocols

- ❑ Two types of Acks:
 - Normal TCP acks
 - Partial acks: Informing source that packet was received by intermediate host, e.g., base station
- ❑ If partial ack is received but the normal ack is not received or DupAcks received, the sender does not back off, simply retransmits
- ❑ Ref: Cobb95, Biaz97

Receiver-Based Scheme

- ❑ Receiver uses inter-arrival time between packets to *guess* the cause of packet loss
- ❑ If the loss is guessed to be due to error, sender is informed via ELN bit in duplicate ack or explicit message
- ❑ Advantages:
 - Can be implemented without base modification
 - Works with encrypted packets
- ❑ Disadvantage:
 - Works only if the wireless link is the slowest link
Ensure that there is some queuing at the base
 - Queuing delays at the base for all packets should be similar
- ❑ Ref: Biaz98

Sender-Based Discrimination Scheme



- ❑ Sender uses roundtrip times, window sizes, and loss pattern to guess the cause of packet loss
- ❑ If loss is guessed to be due to errors, the sender does not back off. Simply retransmits.
- ❑ Heuristics:
 - Delay gradient $dD/dW > 0 \Rightarrow$ Congestion
 - Throughput gradient $dT/dW < 0.5 \Rightarrow$ Congestion
 - $W/RTT_{min} - W/RTT_{actual} > b \Rightarrow$ Congestion
- ❑ Disadvantages: Does not work in practice.
Delay and throughput measurements are quite noisy.
- ❑ Ref: Biaz98b, Biaz99b

TCP Over Satellite

- ❑ IETF TCPSAT
- ❑ Satellite Transport Protocol (STP)
- ❑ Early Acks: ACKprime
- ❑ Our results

IETF TCPSAT

- ❑ Large propagation delays => Large bandwidth delay product
=> Large windows => Use window scaling option

$$\text{Window} = \text{window} * 2^{\text{Scaling factor}}$$

- ❑ Use Selective acknowledgements
=> Allows multiple packets to be recovered in one RTT
- ❑ Do not delay acks => Ack every packet
- ❑ Use larger initial window size (suggested 4kB)
- ❑ Byte Counting: Increase window by number of bytes acked rather than just 1 MSS per RTT
- ❑ Reduce bursts from the sender
- ❑ Ref: RFC 2488, 2760

Satellite Transport Protocol (STP)

- ❑ Non-TCP transport protocol
- ❑ No retransmission timer
- ❑ Sender periodically requests receiver to ack received packets => Save reverse bandwidth if no errors
- ❑ Receiver can also send “Selective Nacks” if packets lost
- ❑ Ref: Henderson98

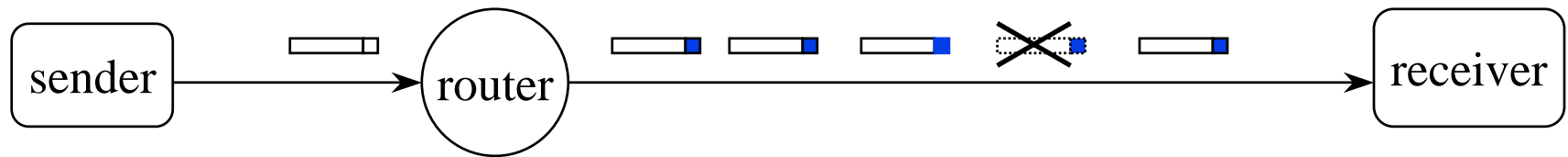
Early Acks: ACKprime

- ❑ Ground stations send partial acks => Grows congestion window
- ❑ Full acks from the receiver required for reliable delivery
- ❑ Ref: Scott98

Our Results for Satellite Networks

- ❑ End System Improvements:
 - Slow start
 - Fast Retransmit and Recovery
 - New Reno
 - SACK
- ❑ Intermediate System Improvements: Drop policies
- ❑ For satellite paths, end system improvements have more impact than intermediate-system based improvements
- ❑ SACK helps significantly
- ❑ Fairness depends upon the drop policies in the intermediate systems and not on end system policies

Wireless Networks: Our Solution



Desired Attributes of the Solution:

1. Must maintain TCP's end-to-end semantics: A packet is acked only after received by the final destination.
2. Modifications must be local: Only Base Station (BS) and Mobile Host (MH) are in the control of wireless service provider. Cannot change all locations that MH visits.
3. Must apply to two-way traffic: MH can be both a sender and a receiver.
4. Wireless links can be at the end or in the middle (satellite links)

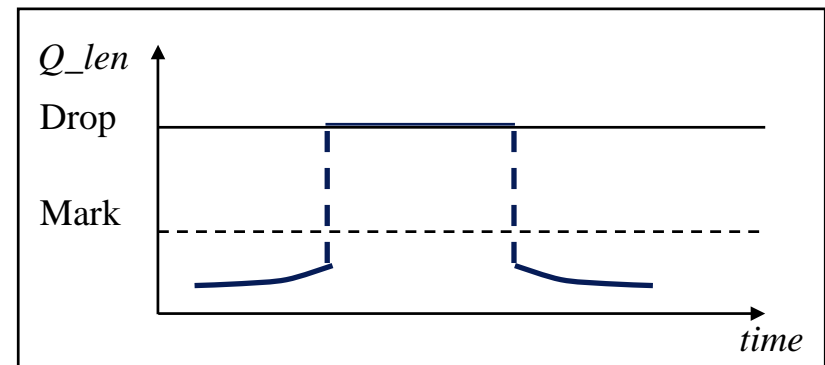
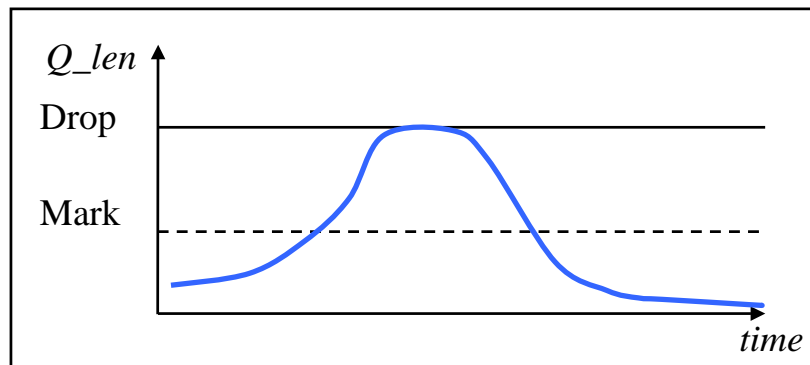
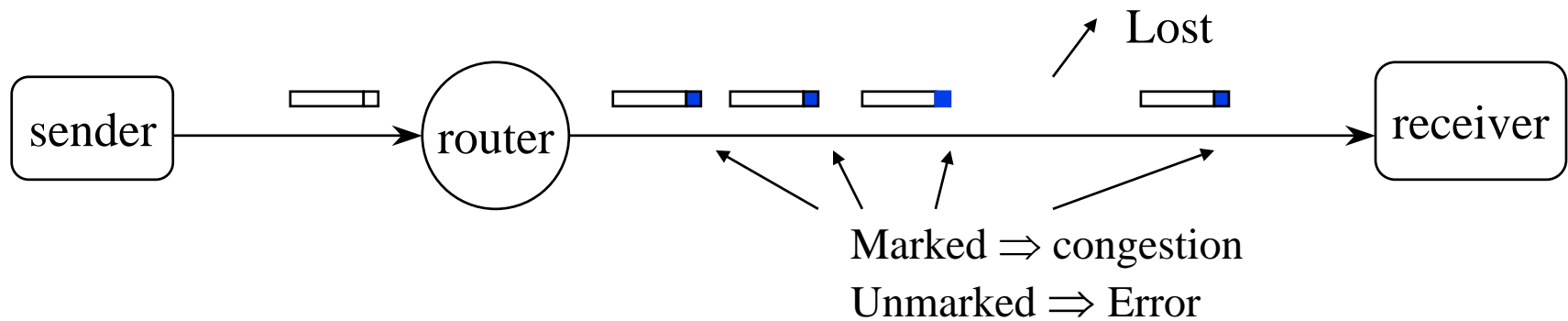
Ref: Liu and Jain 2003

Survey of Prior Proposals

	I-TCP	Multiple Acks	Control Connection	Snoop	ELN	Delayed Dupacks
Semantic	No	Yes	Yes	Yes	Yes	Yes
Local	Yes	No	No	Yes	Yes	Yes
Encryption	No	No	Yes	No	No	Yes
Two-way	Yes	No	No	No	No	No
Intermediate links	Yes	No	Yes	No	No	Yes

Congestion Coherence

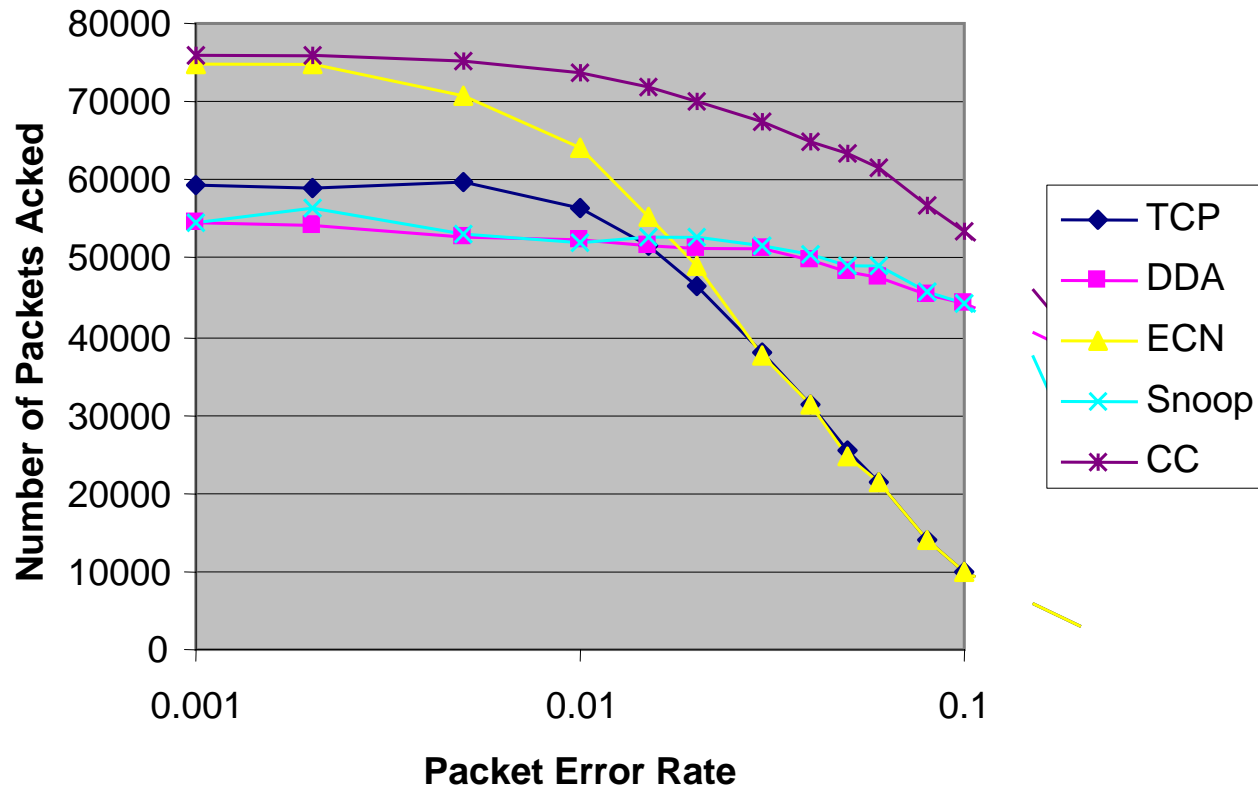
- ❑ Congestion does not happen nor disappear suddenly:
 - Before congestion reaches the point where a packet has to be dropped, some packets must have been marked.
 - After a packet is lost, some packets will be marked.



Congestion Coherence Algorithm

- ❑ Link layer acks and retransmissions at all wireless nodes.
- ❑ **Receiver:**
 - Out-of-order packets received check ECN bits.
 - If any packet marked, send duplicate acks Otherwise, defer the duplicate acks.
 - If expected packet arrives, drop deferred dupacks.
 - If the packet times out, release all deferred dupacks.
- ❑ **Sender:**
 - When the third duplicate acks arrives, MH checks the ECN-ECHO bits.
 - If any of thee duplicate acks carry an ECN-ECHO, MH retransmits the lost packet and reduces the window. Otherwise, TCP defers the retransmission.
 - When the expected ack arrives, cancel the deferred retransmission.
 - If the expected ack does not arrive in certain period of time then MH starts the deferred retransmission.

Goodput



❑ Congestion Coherence provides the highest throughput

Summary



- ❑ Frequent errors on wireless links trigger the congestion mechanism in TCP resulting in low throughput
- ❑ Key mechanisms are link level schemes to reduce/hide error losses, split TCP, TCP modification in base, receiver, or sender
- ❑ Since congestion builds up slowly, coherence of ECN bits provides a good distinction of congestion vs. errors
- ❑ On satellite links, window scaling, large initial windows, and SACK are helpful

Reading Assignment

- ❑ Read sections 9.1 through 9.6 of Murthy and Manoj
- ❑ See also references at the end of this presentation

Homework

- ❑ **Exercise:** A TCP entity opens a connection and uses slow start. Approximately how many round-trip times are required before TCP can send N segments.
- ❑ **Hint:** Write down what the CWND and total segments will be after 1 round trips, 2 round trips, 3 round trips, ...

References

- ❑ C. Liu and R. Jain, "Approaches of Wireless TCP Enhancement and A New Proposal Based on Congestion Coherence", the 36th Hawaii International Conference on System Sciences, Quality of Service in Mobile and Wireless Network minitrack, Big Island, Hawaii, January 5-9, 2003, pp. 307a, <http://www.cse.wustl.edu/~jain/papers/hicss.htm>
This paper also has references to several other papers on wireless.
- ❑ Nitin Vaidya, "TCP for Mobile and Wireless Hosts," a comprehensive tutorial presentation, 291 pp., <http://www.cyon.org.uk/wp-content/uploads/2005/12/tcp-wireless-tutorial.ppt>