

無線網路多媒體系統

Wireless Multimedia System

Lecture 8: Wireless TCP

吳曉光博士

<http://inrg.csie.ntu.edu.tw/wms>

*We
provide
Wireless
Wireless Network & Multimedia Laboratory
Solution*

Agenda

- ◆ Basic TCP
- ◆ Impact of Mobility & Wireless on TCP performances
- ◆ Solutions for Wireless TCP
- ◆ Midterm (next week)

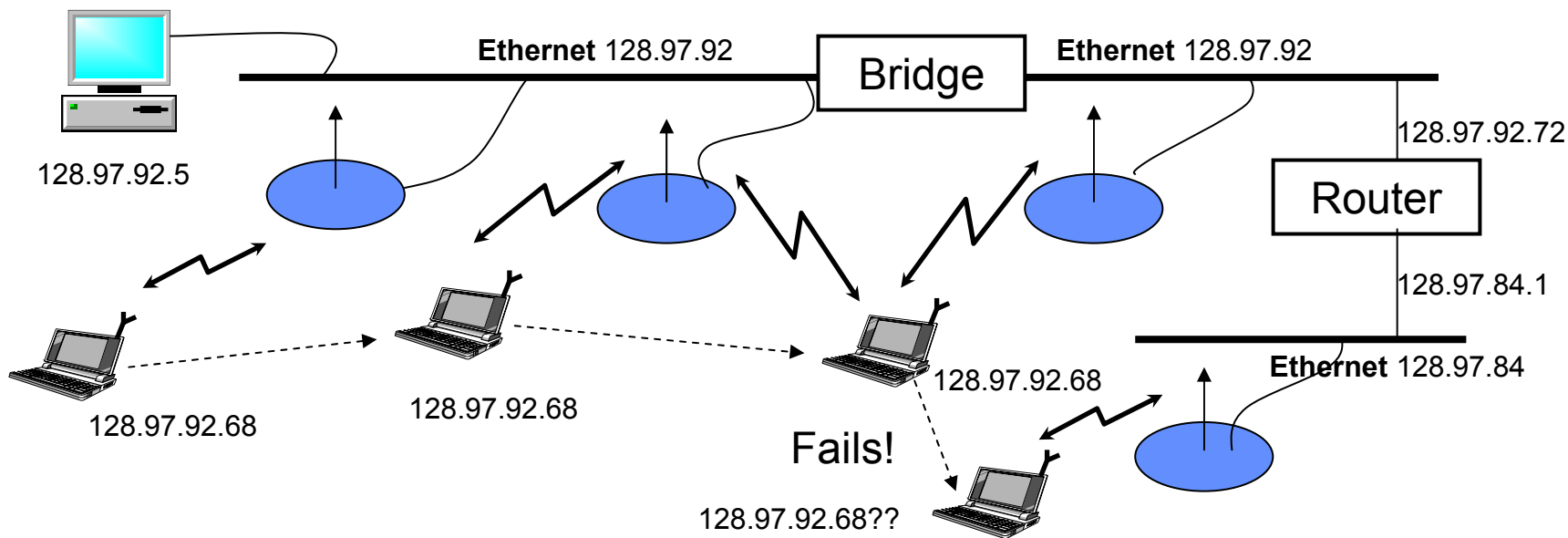


Reading

- ◆ [Balakrishnan95], Harri Balakrishnan, Srinivasan Seshan, Elan Amir and Randy H. Katz, “Improving TCP/IP Performance over Wireless Networks”, ACM Mobicom95
- ◆ [Balarkrishnan97], Harri Balarkrishna, Venkat N, Padmanabhan, Srinivasan Seshan and Randy Katz, “A Comparison of Mechanisms for Improving TCP Performance over Wireless Links”, IEEE JSAC 97.
- ◆ Reference: [Mario2001], Saverio Mascolo, Claudio Casetti, Mario Gerla, Renwang “TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links”, Mobicom2001

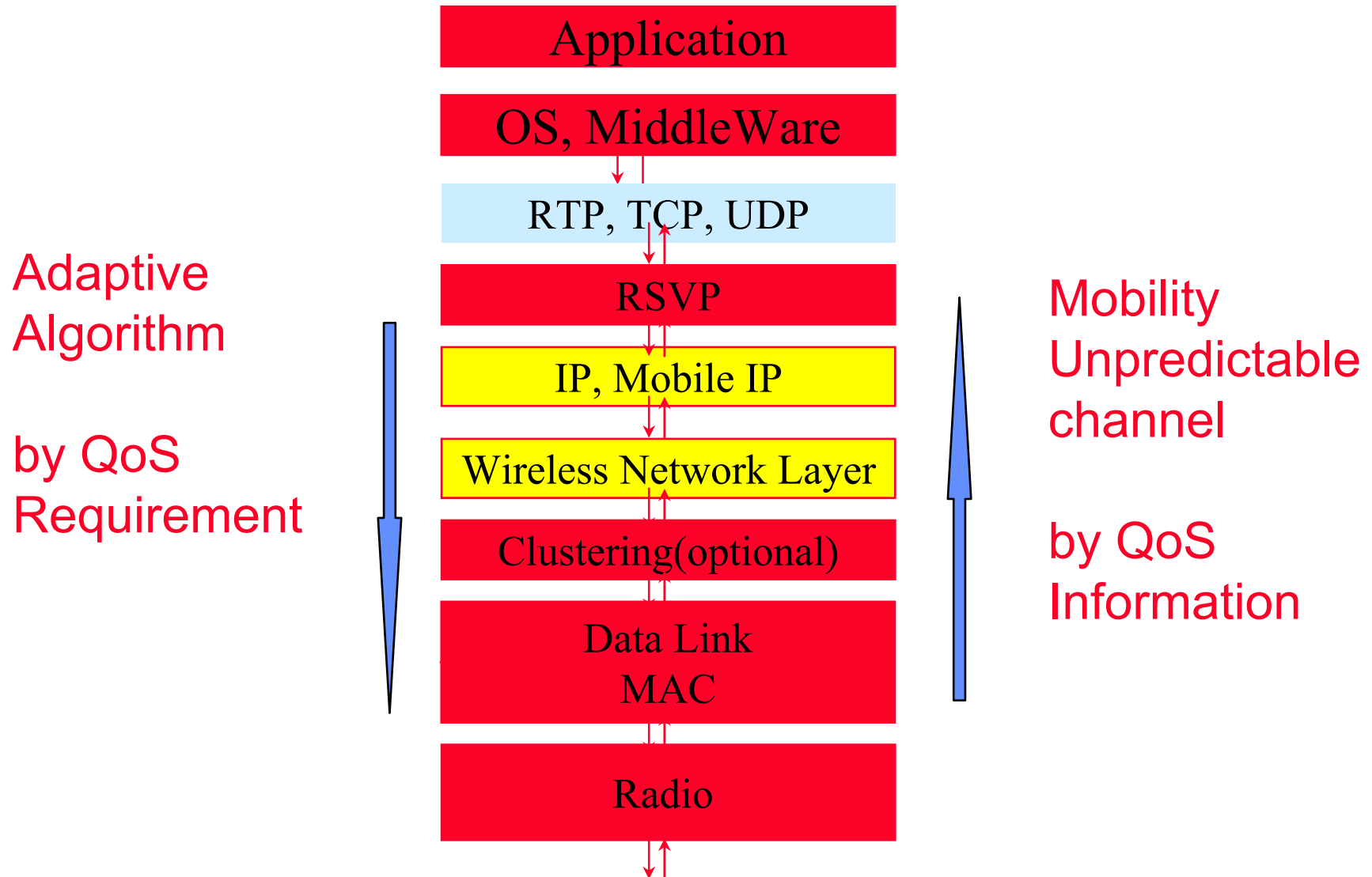


Mobility in Wireless LANs: Basestation as Bridges



- ◆ Basestations are bridges(layer 2) – i.e. they relay MAC frames
 - Smart bridges avoid wasted bandwidth
- ◆ Works the within an ethernet(or other broadcast LAN)
 - Fails across network boundaries, and in switched LANs(e.g. ATM)

QoS and Multimedia Traffic Support



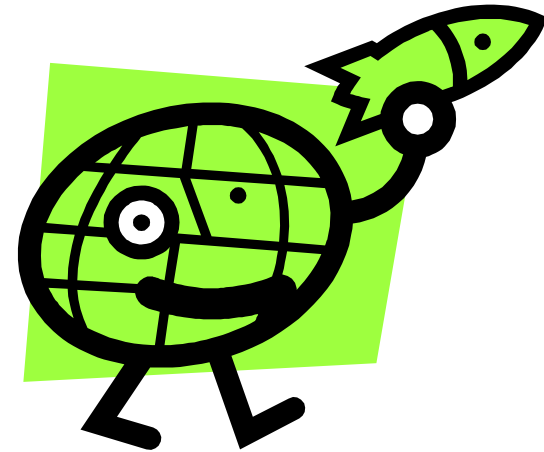
Background

- ◆ With the growth of wireless device, wireless network access will become popular, but...
- ◆ Import the protocol from the wire network to wireless network...
- ◆ Packet losses occur in wireless due to the lossy links, not network congestion
- ◆ In traditional TCP, it can not distinguish the difference between that

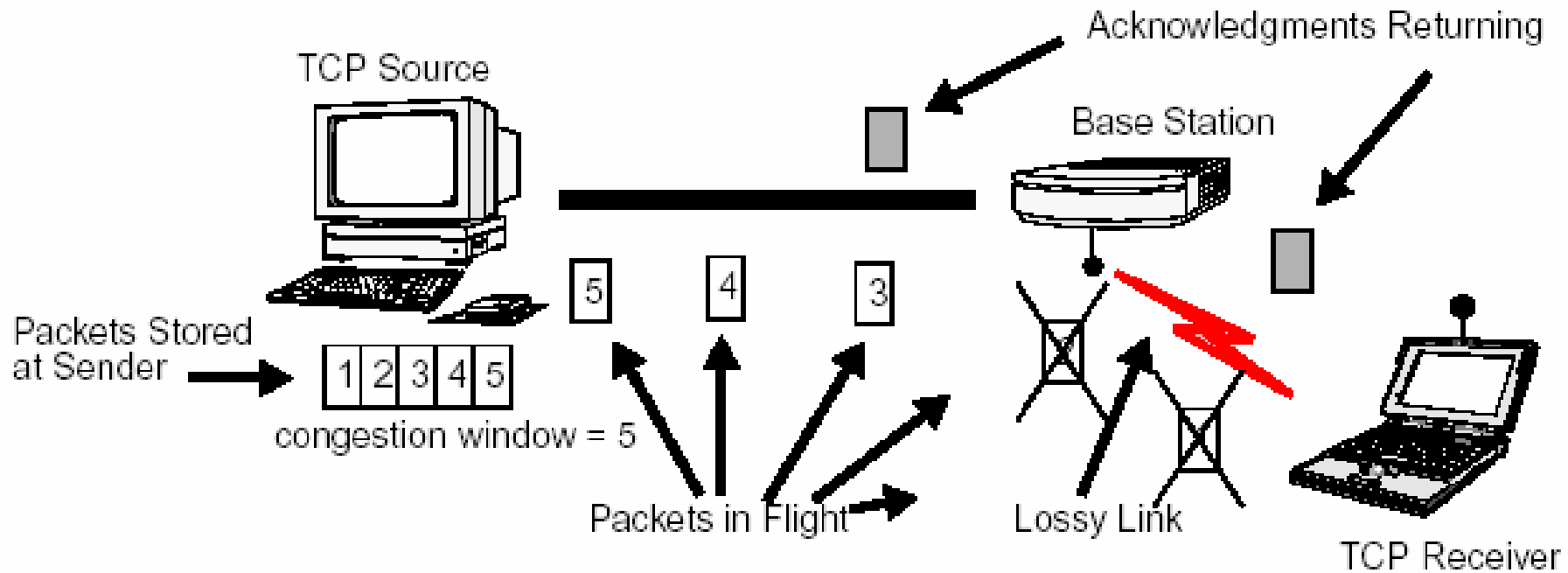
Characteristics of Wireless & Mobility

- ◆ Limited Bandwidth
 - Small frame sizes (MTU) to keep latency small
- ◆ High bit error rates
 - Small frame sized to keep packet loss probability small
- ◆ Time varying bit error rate
 - Fading, frequency collisions etc.
- ◆ QoS (loss rate, delay) degradation during hand-off
 - Due to network layer rerouting
 - Due to link layer procedures
- ◆ QoS degradation after hand-offs
 - Lack of resource at new basestation
 - Less optimal route

Basic End-to-End Control (Transport)



Typical loss situation



UDP (Connectionless, Unreliable)



Possible Multicast, Real Time Traffic, TCP-Friendly

Impact on Connectionless, Unreliable Transport Protocol



- ◆ Example: effect on UDP applications
- ◆ Increase in end-to-end packet losses
 - Error on wireless link
 - Packet loss during hand-offs
- ◆ Drop in application throughput
 - Errors on wireless link
 - Packet loss during hand-off
- ◆ Pauses in interactive applications
 - Burst errors on wireless link
 - Packet loss during hand-off
 - Delay increase due to buffering & re-sequencing during hand-offs
- ◆ Application level impact is much more complex!

TCP (Connection Oriented, Reliable)



Data Transmission, WWW, flow control, error control

TCP Basics

- ◆ Sliding window protocol: Go-Back N ARQ
 - Transfers a byte stream in “segments”, not fixed user blocks, logical timer associated with each segment that is sent
 - 32-bit sequence number indicated byte number in stream
 - ◆ Window is max number of outstanding unACK'ed bytes in network
- ◆ Cumulative acknowledgement scheme (original TCP)
 - Ack's all bytes up through n
 - Piggybacked on data packets in reverse direction
- ◆ Control of sender's window size
 - Min (receiver's advertized window, congestion window)
 - Three goals
 - ◆ Flow control to avoid receiver buffer overflow
 - ◆ Congestion control to react to congestion in network layer & below
 - ◆ Congestion avoidance
- ◆ Segment loss is assumed to be a result of congestion in routers
 - Reasonable for wired network since BER on fiber is better than 10^{-12}

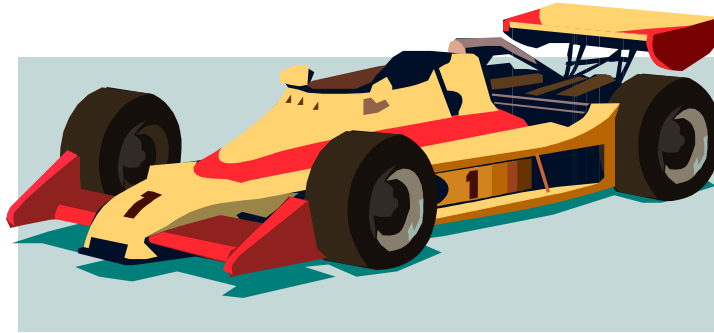
TCP's End to End Congestion Control

- ◆ Window-based congestion control
 - Cwnd: congestion window size
 - Ssthresh: slow start threshold (for slow down of increase)
- ◆ Timeout is an indicator of segment loss
- ◆ Timeout value
 - Using estimated average of ACK delay and expected deviation
- ◆ On timeout
 - Segment is assumed lost and is attributed to congestion
 - One-half of current window is recorded in ssthresh
 - Cwd is reduced to 1
 - Timeout value is increased in case packet was delayed

TCP's End-to-end Congestion Control

- ◆ On new ACK
 - Everything okay, so allow larger congestion window
 - Two ways of increasing cwnd
 - ◆ Phase1: slow start until $cwnd \leq ssthresh$
 - Fast (exponential) increase of cwnd
 - ◆ Phase2: congestion avoidance
 - Slow (additive) increase of cwnd
- ◆ Duplicate ACKs
 - Two causes: lost segment, misordered segment
 - ≥ 3 duplicate ACKs in a row are a good indication of a lost segment but data is still flowing
 - Fast Retransmit and Fast Recovery
 - ◆ Missing segment is retransmitted without waiting for timeout
 - ◆ One half of current window is recorded in ssthresh
 - ◆ Congestion avoidance is done but not slow start

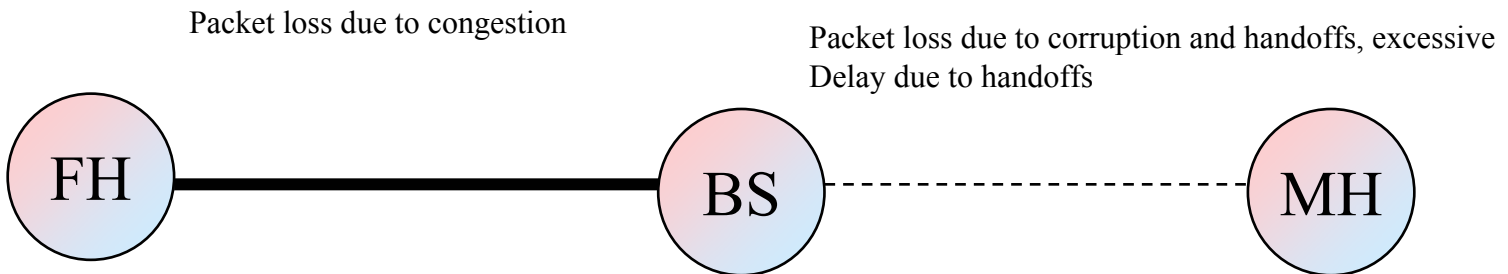
Challenges of Mobility and Wireless on Network Performance



TCP Performance

The Problem

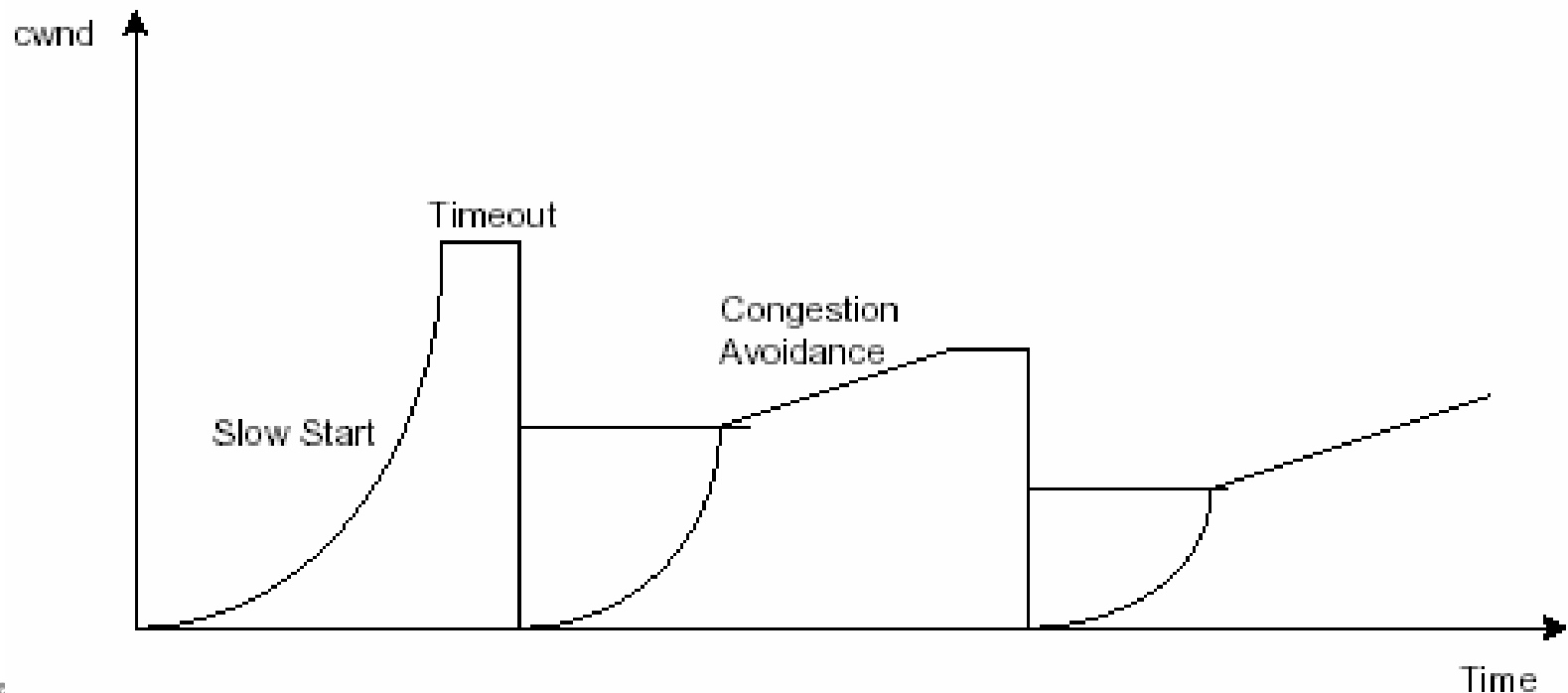
- ◆ In Wireless and mobile networks, segment loss is likely not due to congestion
 - Packet corruption due to high BER on wireless link (noise, fading)
 - Packet delay and losses during handoffs
- ◆ But, TCP invokes congestion control nevertheless
- ◆ Mistaking wireless errors and handoffs for congestion causes
 - Significant reductions in throughput (window size decreases, slow start)
 - Unacceptable delays (low resolution TCP times ~500ms, back-off)



Example graph

$cwnd \leq ssthresh \rightarrow$ slow start

$cwnd > ssthresh \rightarrow$ congestion avoidance

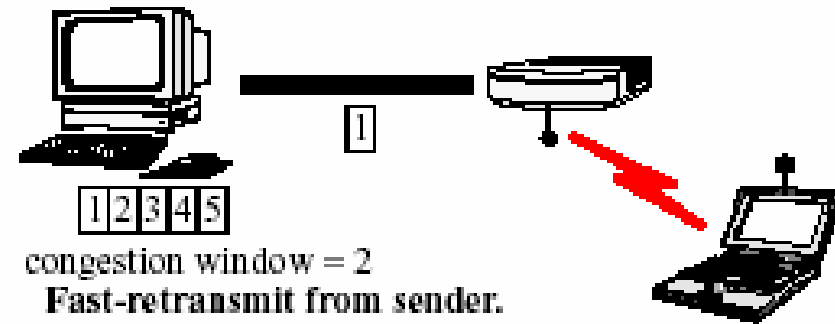
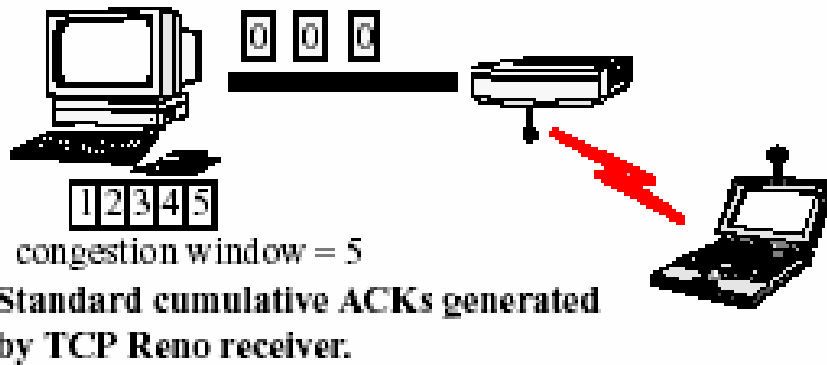


Fixes?

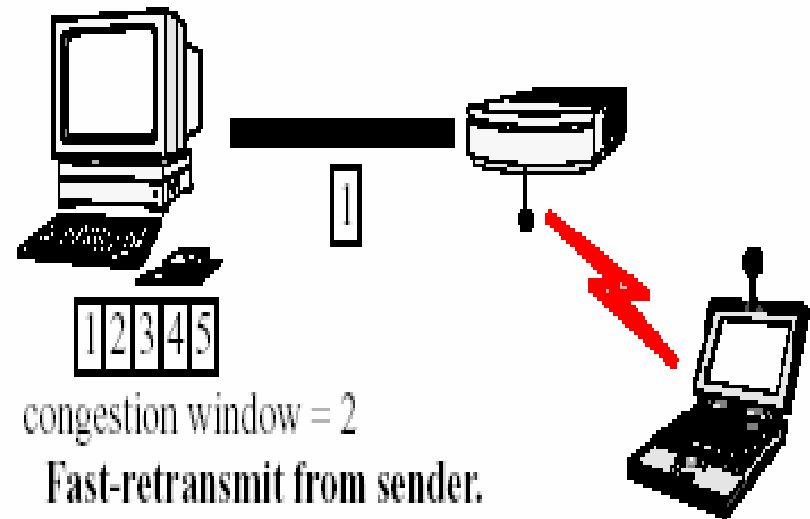
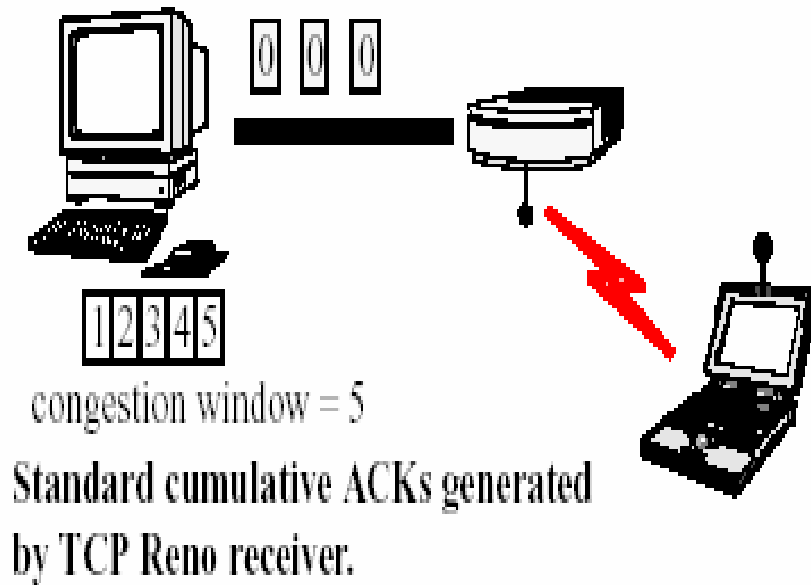
- ◆ Fix TCP
 - TCP really a hack in many ways..
 - Separate congestion control from error control
 - Move away from cumulative ACK
- ◆ Fix lower layer to make TCP work better
 - Improve the wireless link
- ◆ Use something different
 - Something totally new
 - Something different for the wireless part



Normal TCP



Fast-Retransmit Scheme



Solutions for WTCP (I)



Split the connection into two parts

Split Connection Approaches

- ◆ Main Idea: split MH \leftrightarrow FH connection into two MH \leftrightarrow BS & BS \leftrightarrow FH
 - Separate flow control and reliable delivery mechanisms
 - Intermediate higher layer agent at the base-station
 - Session layer hides the split connection
- ◆ Two approaches:
 - Both FH \leftrightarrow BS & BS \leftrightarrow MH segments use TCP: Rutgers' Indirect-TCP
 - ◆ e.g. uses MTCP (Multiple TCP) over BS \leftrightarrow MH
 - BS \leftrightarrow MH uses specialized protocol
 - ◆ e.g. uses SRP (Selective Repeat) over BS \leftrightarrow MH
 - ◆ Error and flow control optimized for lossy wireless link

Pros & Cons of Split-Connection Approaches

◆ Pros

- FH is shielded from wireless link behavior
- Handoff is transparent to FH
- Relative easy to implement
- Requires no modification to FH
- Can use specialized protocol over wireless link

◆ Cons

- Loss of end-to-end semantics
- Application relink with new library
- Software overhead: efficiency and latency
- Large handoff latency

Solutions for WTCP (II)



Lower layer to make TCP work better

Link-level Error Control

- ◆ FEC and ARQ on wireless link to increase its reliability
 - Improves performance independent of transport protocol
- ◆ Disadvantage
 - Coupling between link level and end-to-end retransmission may lead to degraded performance at high error rates
 - Does not address the delay and losses due to handoffs

Solutions for WTCP (III)

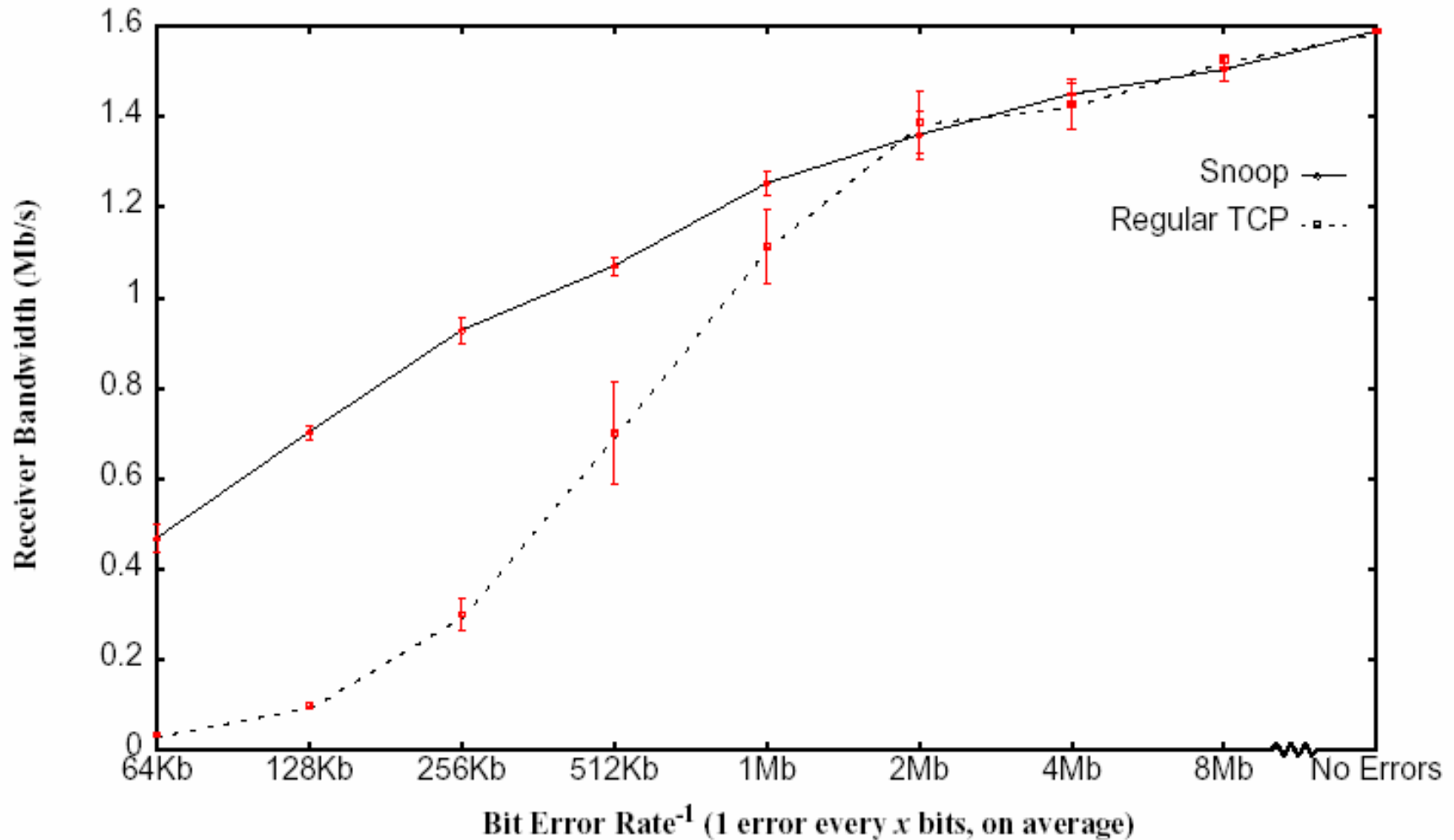


Snoop, Make it look like!

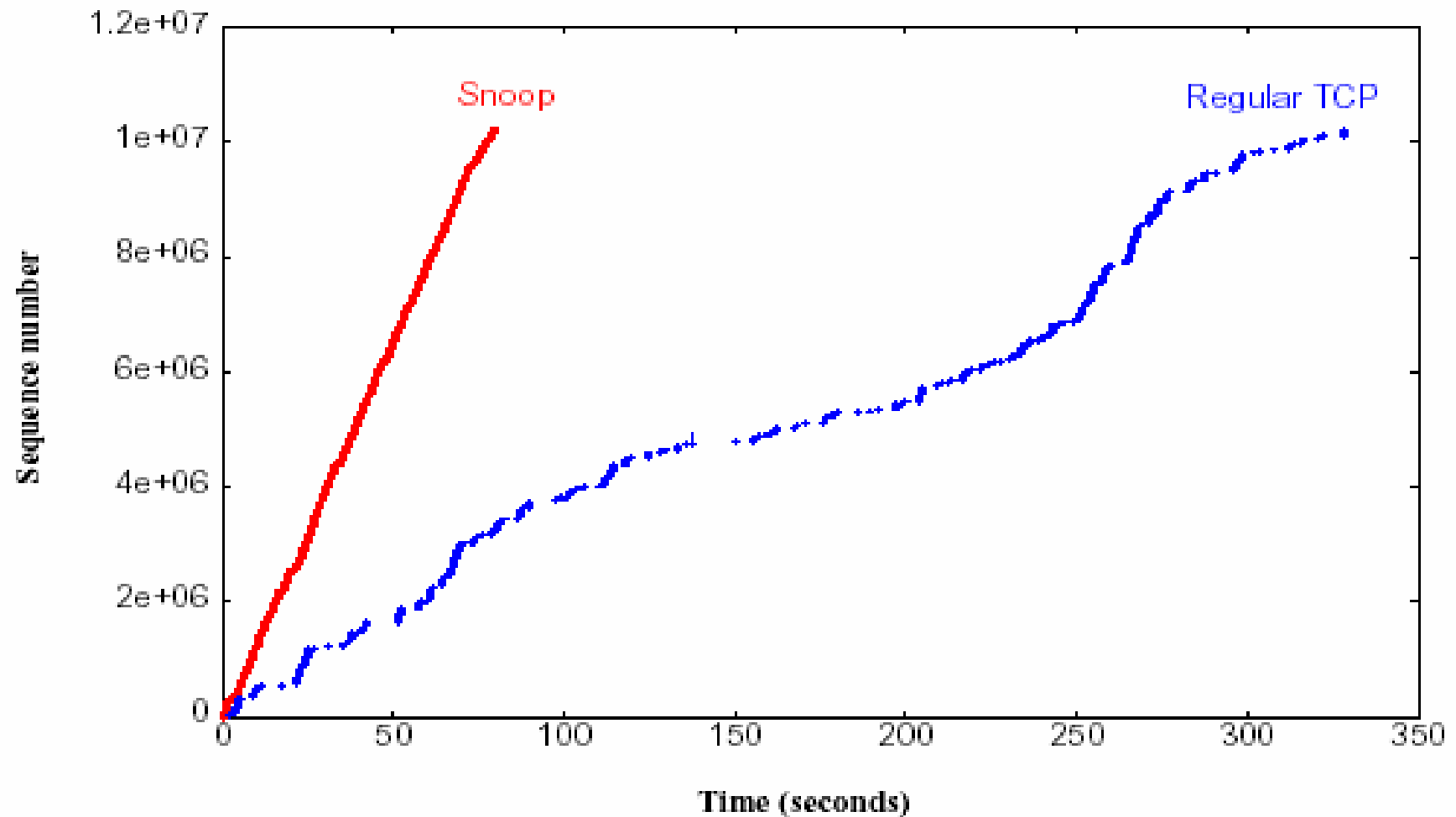
Snoop TCP

- ◆ Basic Idea for transfer of data to MH
 - Snoop Module: Modify network layer routing code at BS
 - Cache un-acknowledged TCP data going to MH at BS
 - Perform local retransmissions over wireless link
 - ◆ Policies to deal with ACKs from MH and timeout
 - ◆ Used duplicate ACKs to identify packet losses
 - Shields sender from wireless link
 - ◆ Transient conditions of high BER, temporary disconnection
- ◆ Basic idea for transfer of data from MH
 - BS detects missing packets and generated NACKs for MH, exploits SACK option for TCP
 - MH re-sends the packets, requires modifying TCP code at MH
- ◆ Features
 - Speedups of up to x20 over regular TCP depending on bit error rate
 - Maintain end-to-end semantics
 - Does not address the handoff problem

Performance of the Snoop Mechanism



Performance of the Snoop Mechanism



Comparison of Wireless TCP Techniques

- ◆ End-to-End proposals
 - Selective ACKs
 - ◆ Allows sender to recover from multiple packet losses without resorting to course timeout
 - Explicit Loss Notification (ELN)
 - ◆ Allow sender to distinguish between congestion vs. other losses
- ◆ Split-connection proposal
 - Separate reliable connection between BS & MH
 - ◆ May use standard TCP or, special techniques such as SACK, or NACK
- ◆ Link-layer proposal
 - Hide link-layer losses via general local retransmission and FEC
 - Make link-layer TCP aware
 - ◆ Snoop agent to suppress duplicate ACKs

Main Conclusions of [Balakrishnan97]

- ◆ Simple link layers do not quite work
 - Adverse interaction of times is actually a minor problem
 - Fast retransmission and associated congestion control gets triggered and cause performance loss
- ◆ Reliable link layer with TCP knowledge works well
 - Shielding sender from duplicate ACKs due to wireless losses improves throughput by 10-30%
- ◆ No need to split end-to-end connections
 - I-TCP does as bad because sender stalls due to buffer space limit at BS
 - Using SAK or BS-MH link works well
- ◆ SACK and ELN helps significantly
 - Help avoid timeous
 - e.g. ELN helped throughput by x2 over vanilla TCP-Reno
 - But still do 15% to 35% worse than TCP-aware link layer schemes

Introduction

- ◆ TCP Westwood (TCPW) is a sender-side modification of TCP Reno in wire as well as wireless network
- ◆ TCPW can estimate the E2E b/w and the improvement is most significant in wireless network with lossy links
- ◆ TCPW sender monitors the ACK reception and from it estimates the data rate
- ◆ The sender uses the b/w estimate to properly set the cwin and ssthresh

Filtering the ACK reception rate

- ◆ Sample of bandwidth

- ◆ We measure $b_k = \frac{d_k}{t_k - t_{k-1}}$ to average sampled