ECS-087: Mobile Computing

TCP over wireless

TCP and mobility

Most of the Slides borrowed from

Prof. Sridhar Iyer's lecture

IIT Bombay

Effect of Mobility on Protocol Stack

- Application: new applications and adaptations
- Transport: congestion and flow control
- Network: addressing and routing
- Link: media access and handoff
- Physical: transmission errors and interference

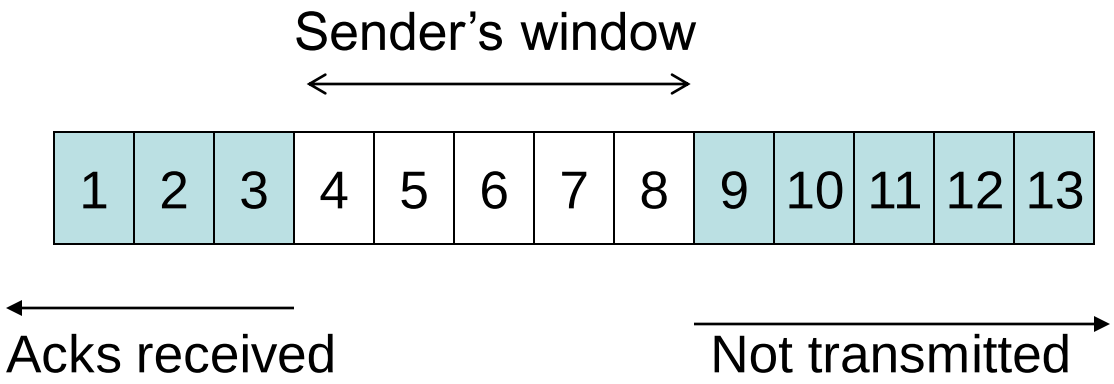
TCP basics

- Reliable, ordered delivery
  - uses sequence numbers, acknowledgements, timeouts and retransmissions
  - End-to-end semantics (ACK *after* data recd)

- Provides flow and congestion control
  - uses sliding window based buffers and feedback from receiver/network to adjust transmission rate

Window based flow control

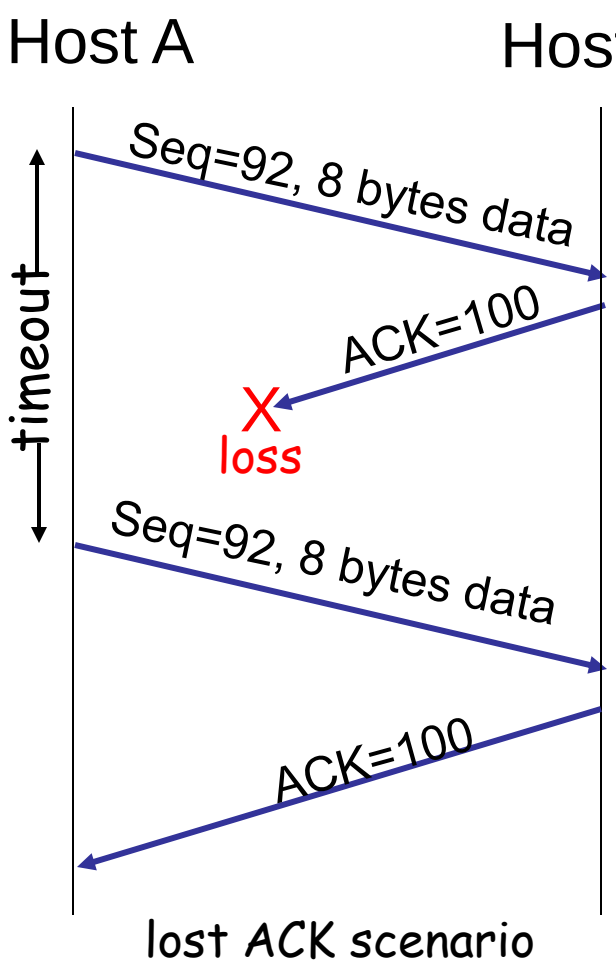
- Window size minimum of
  - receiver's advertised window - determined by available buffer space at the receiver
  - congestion window - determined by sender, based on network feedback



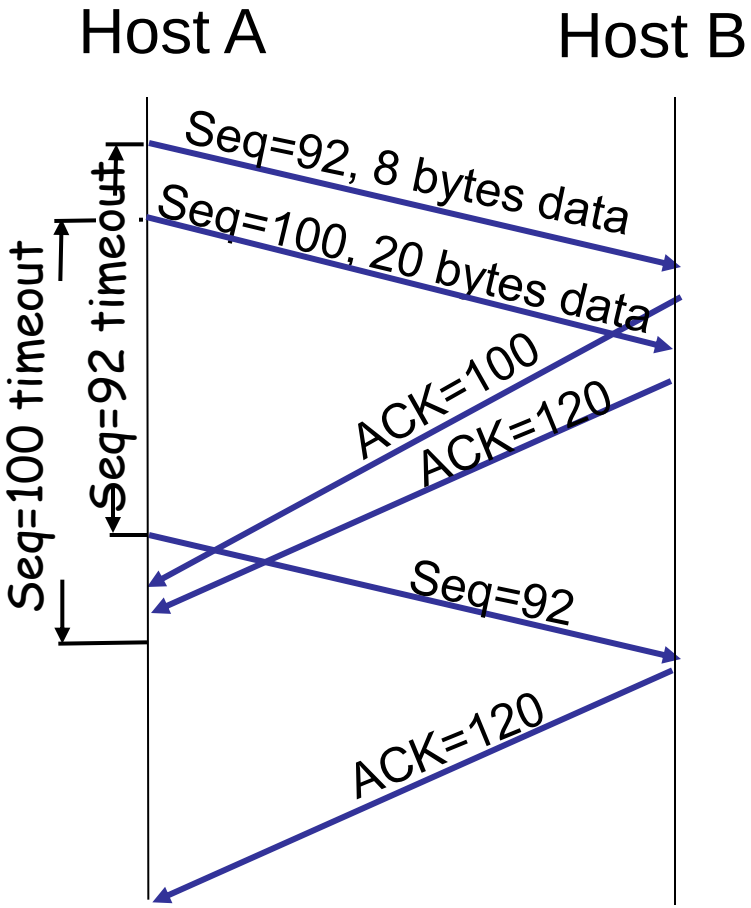
Timeouts and retransmission

- TCP manages four different timers for each connection
  - retransmission timer: when awaiting ACK
  - persist timer: keeps window size information flowing
  - keepalive timer: when other end crashes or reboots
  - 2MSL timer: for the TIME_WAIT state

TCP: retransmission scenarios



lost ACK scenario



premature timeout, cumulative ACKs

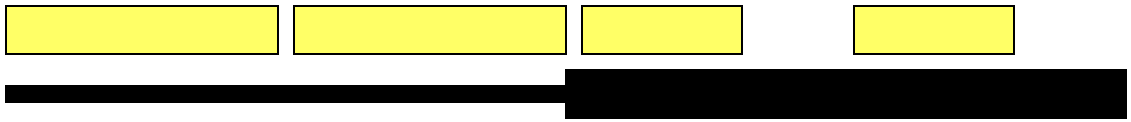
RTT estimation

Exponential Averaging Filter:

- Measure SampleRTT for segment/ACK pair
- Compute weighted average of RTT
  - EstimatedRTT = α PrevEstimatedRTT + (1 – α) SampleRTT
  - RTO = β * EstimatedRTT
- Typically $\alpha = 0.9$; $\beta = 2$

Ideal window size

- Ideal size = delay * bandwidth
  - delay-bandwidth product



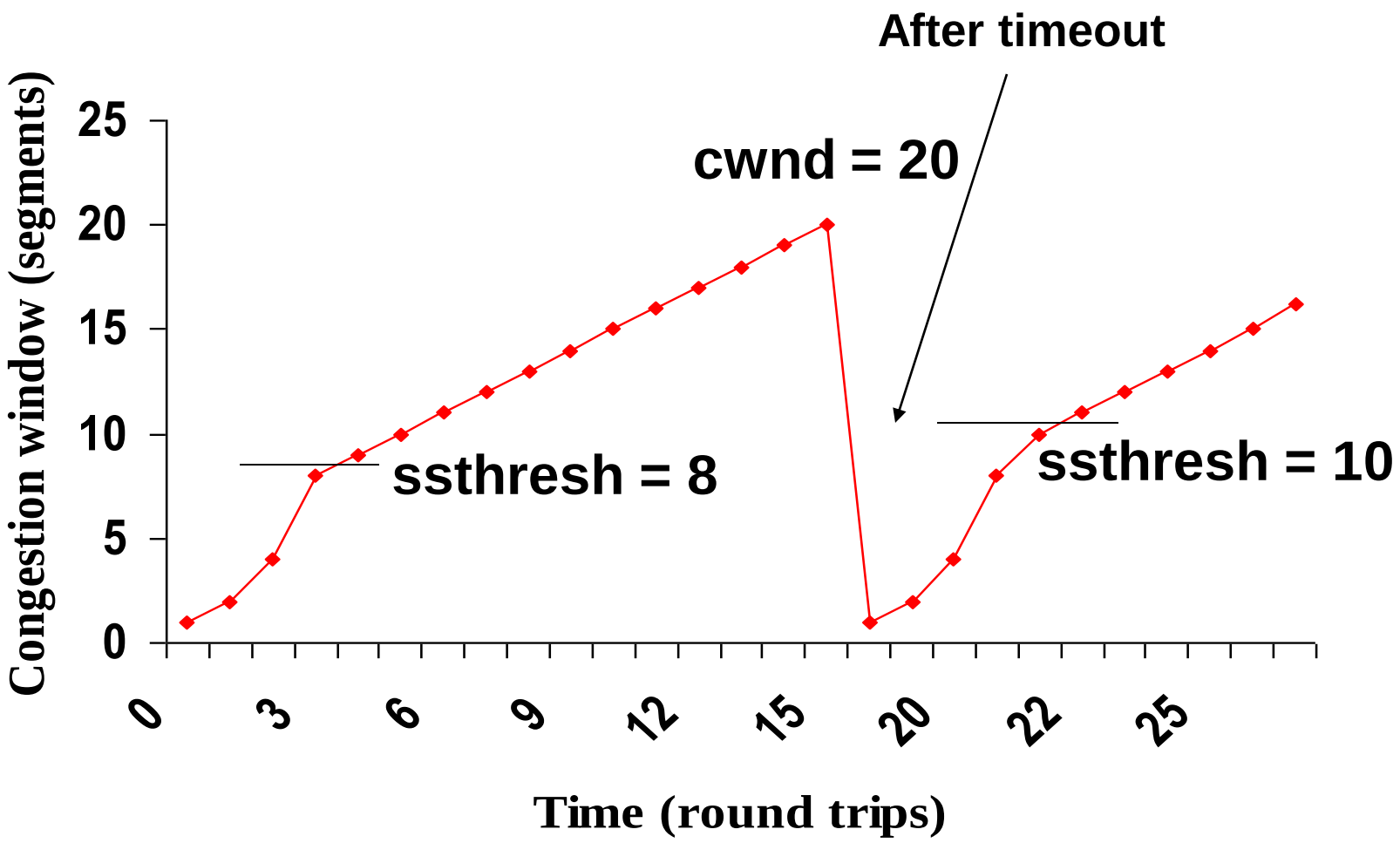
- If window size < delay*bw
  - Inefficiency (wasted bandwidth)
- If window size > delay*bw
  - Queuing at intermediate routers (increased RTT)
  - Potentially, packet loss

Congestion control

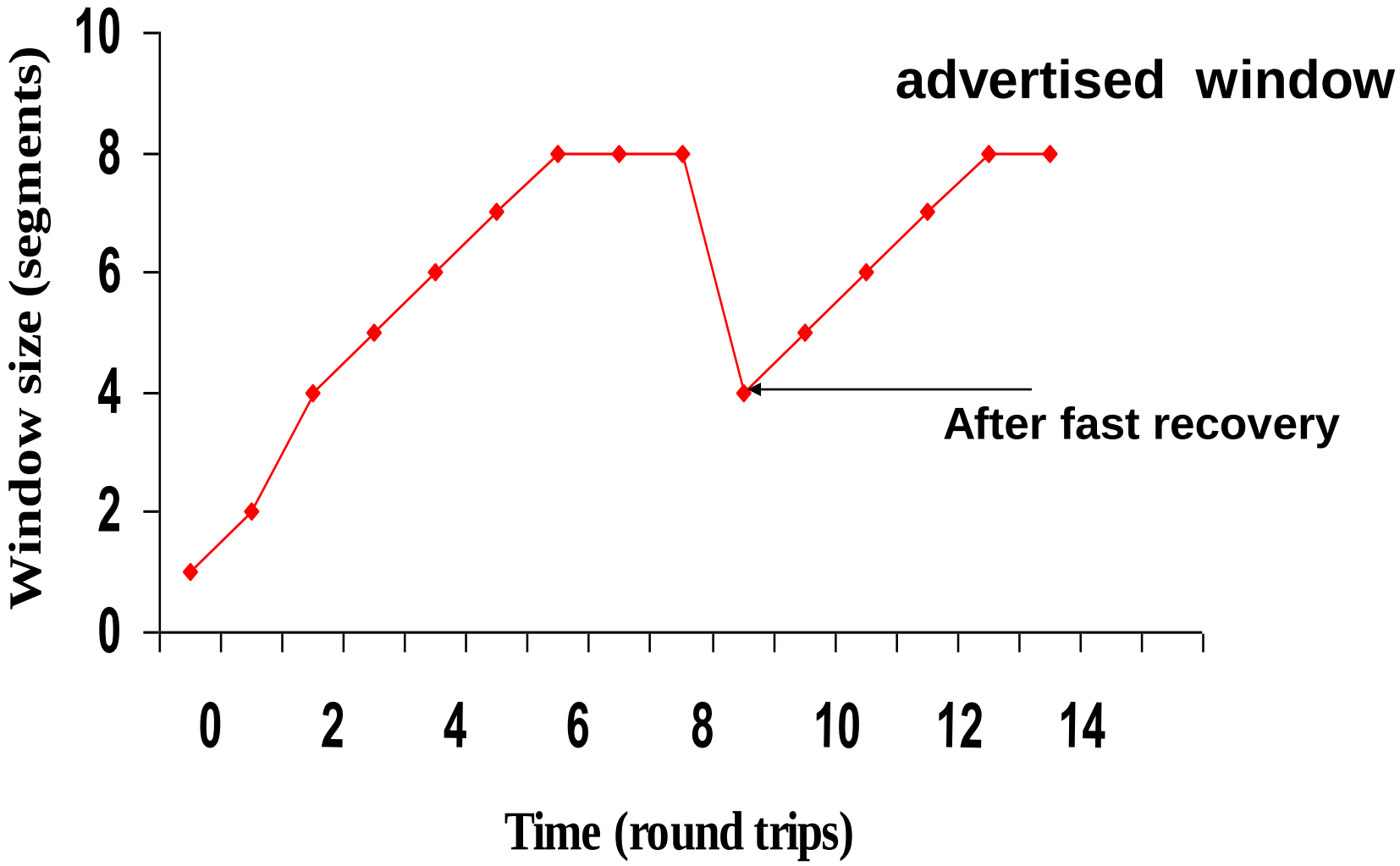
- On detecting a packet loss, TCP sender assumes that network congestion has occurred
- On detecting packet loss, TCP sender drastically reduces the congestion window

- Reducing congestion window reduces amount of data that can be sent per RTT

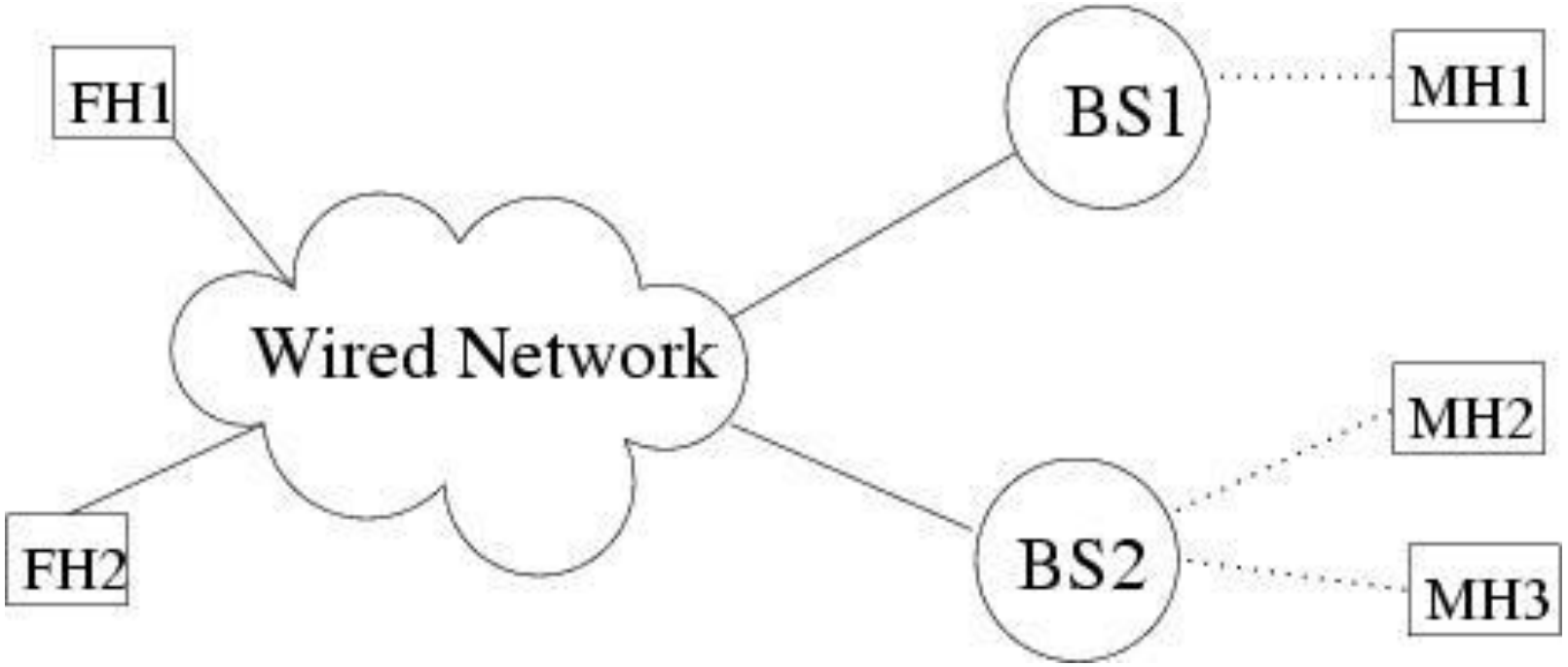
Typical TCP behaviour



Fast retransmit and Fast recovery



Typical mobile wireless scenario



- FH: Fixed Host
- MH: Mobile Host
- BS: Base Station (gateway)

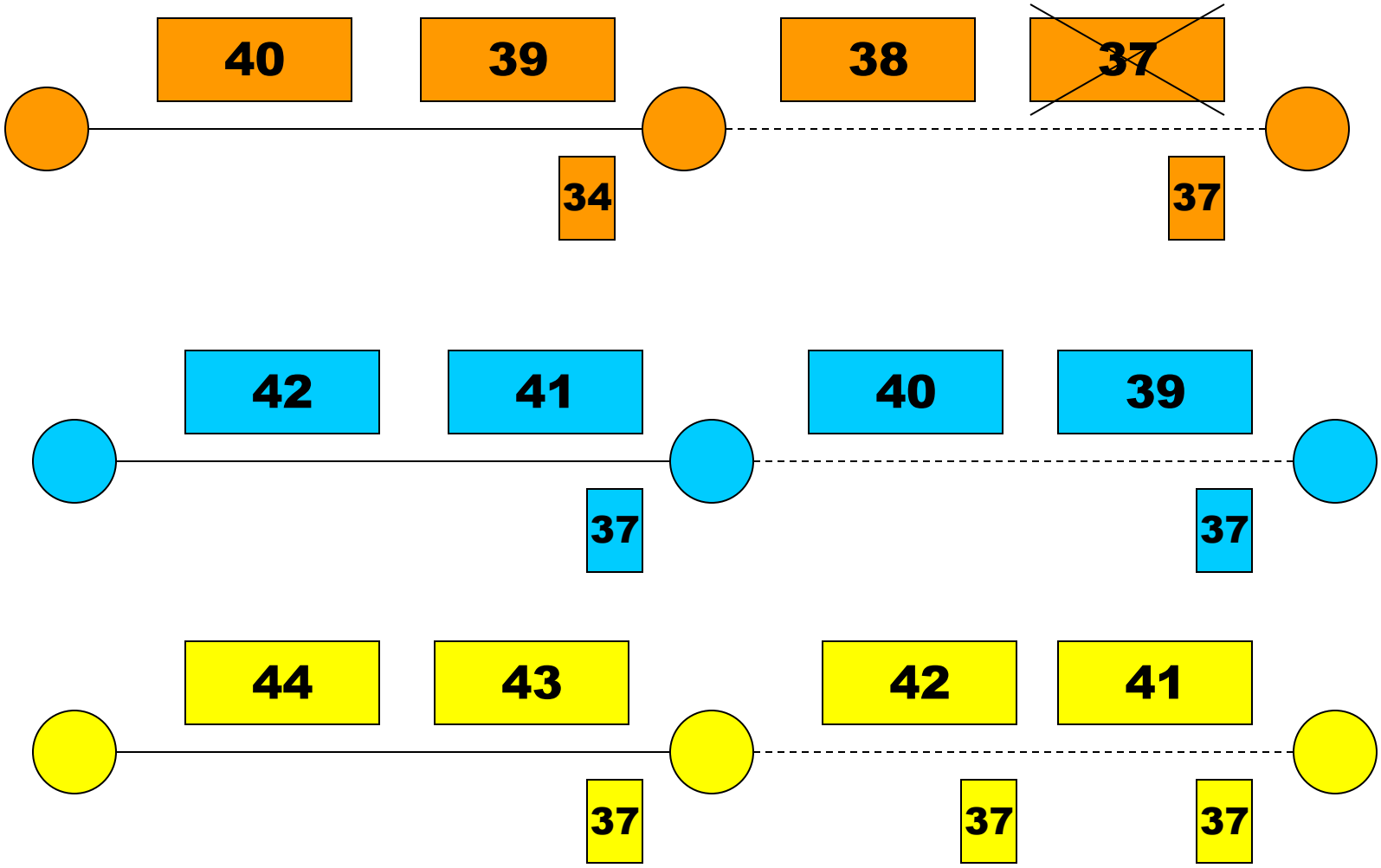
Burst errors may cause Timeouts

- If wireless link remains unavailable for extended duration, a window worth of data may be lost
  - driving through a tunnel; passing a truck
- Timeout results in slow start
  - Slow start reduces congestion window to 1 MSS, reducing throughput
- Reduction in window in response to errors unnecessary

Random errors may cause Fast Retransmit or Timeout

- If a packet is lost due to transient link conditions
  - Channel noise leading to CRC error
- Fast retransmit results in fast recovery
  - Fast recovery reduces congestion window to 1/2
- If multiple packets losses happen in a window,
  - Results in timeout
- Reduction in window in response to errors unnecessary

Example: Random errors



TCP and wireless/mobility

TCP assumes congestion if packets dropped

- typically wrong in wireless networks
  - often packet loss due to *transmission errors*
- *mobility* itself can cause packet loss
  - nodes roam from one access point or foreign agent to another with packets in transit

Motivation for TCP adaptation

Performance of an unchanged TCP degrades severely for wireless/mobile environments

- TCP cannot be changed fundamentally
  - Widely deployed in the fixed network
  - Internet interoperability requirement

- TCP for wireless/mobility has to be compatible with “standard” TCP

Adaptation for TCP over wireless

Several proposals to adapt TCP to wireless environments

- Modifications to TCP implementation at
  - Fixed Host
  - Base Station
  - Mobile Host
- Approaches
  - Hide error losses from the sender
  - Let sender know the cause of packet loss

Ideal behavior

- Ideal TCP behavior: TCP sender should simply retransmit a packet lost due to transmission errors, without taking any congestion control actions
  - Ideal TCP typically not realizable

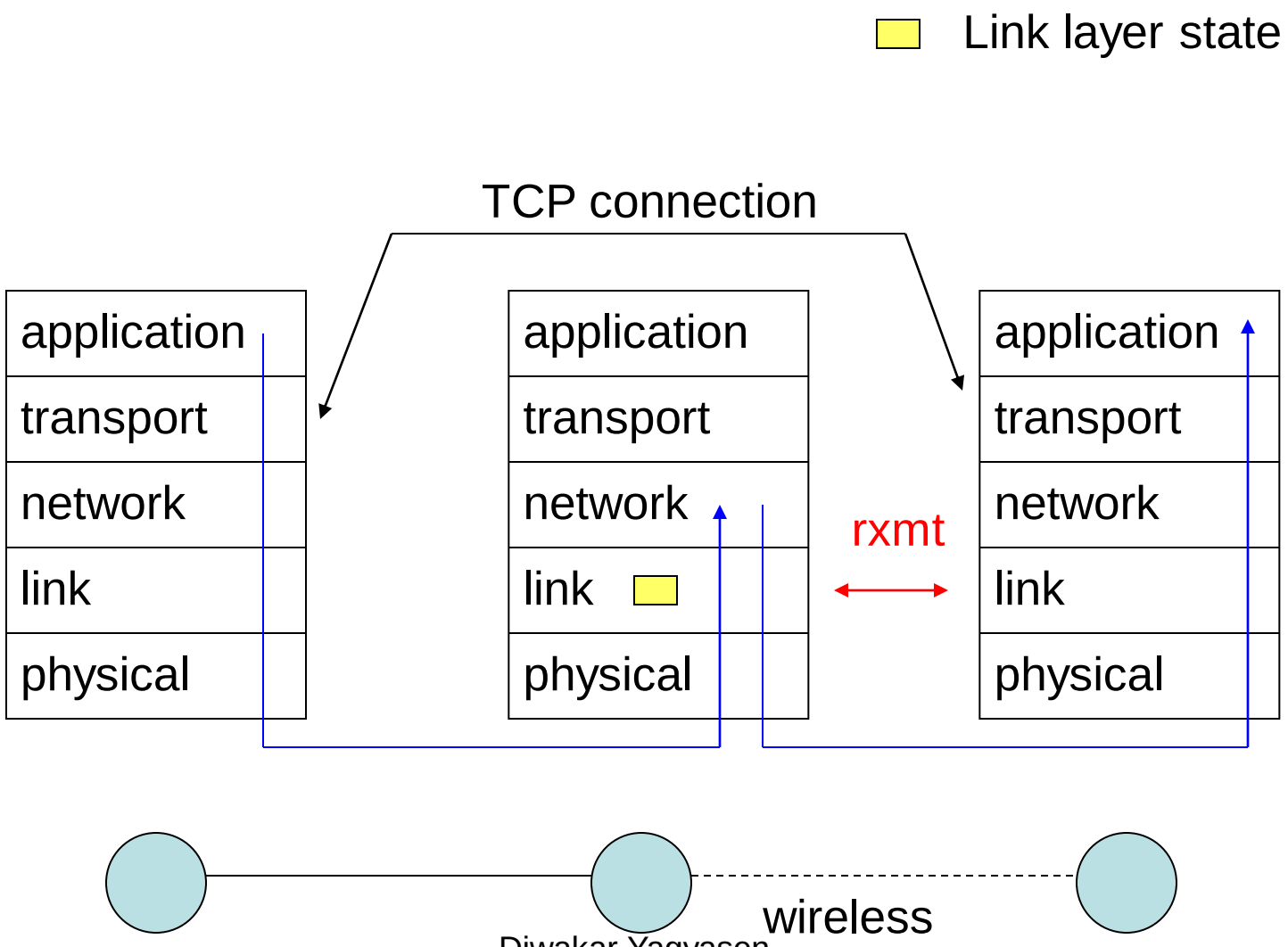
- Ideal network behavior: Transmission errors should be hidden from the sender
  - Errors should be recovered transparently and efficiently
- Proposed schemes attempt to approximate one of the above two ideals

Link Layer mechanisms

- Forward Error Correction (FEC)
  - Can be use to correct small number of errors
  - Incurs overhead even when errors do not occur

- Link Level Retransmissions
  - Retransmit a packet at the link layer, if errors are detected
  - Retransmission overhead incurred only if errors occur

Link Level Retransmissions

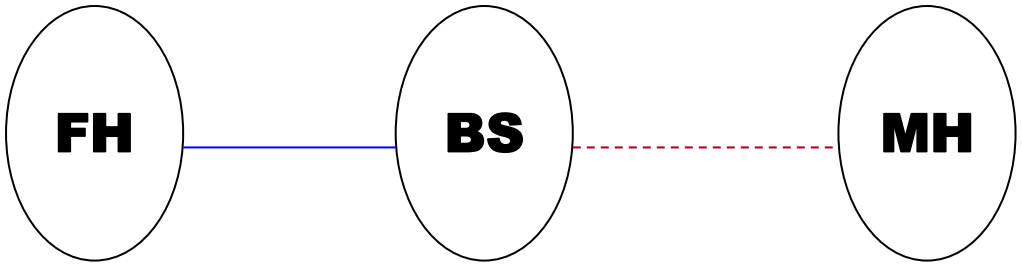


Issues

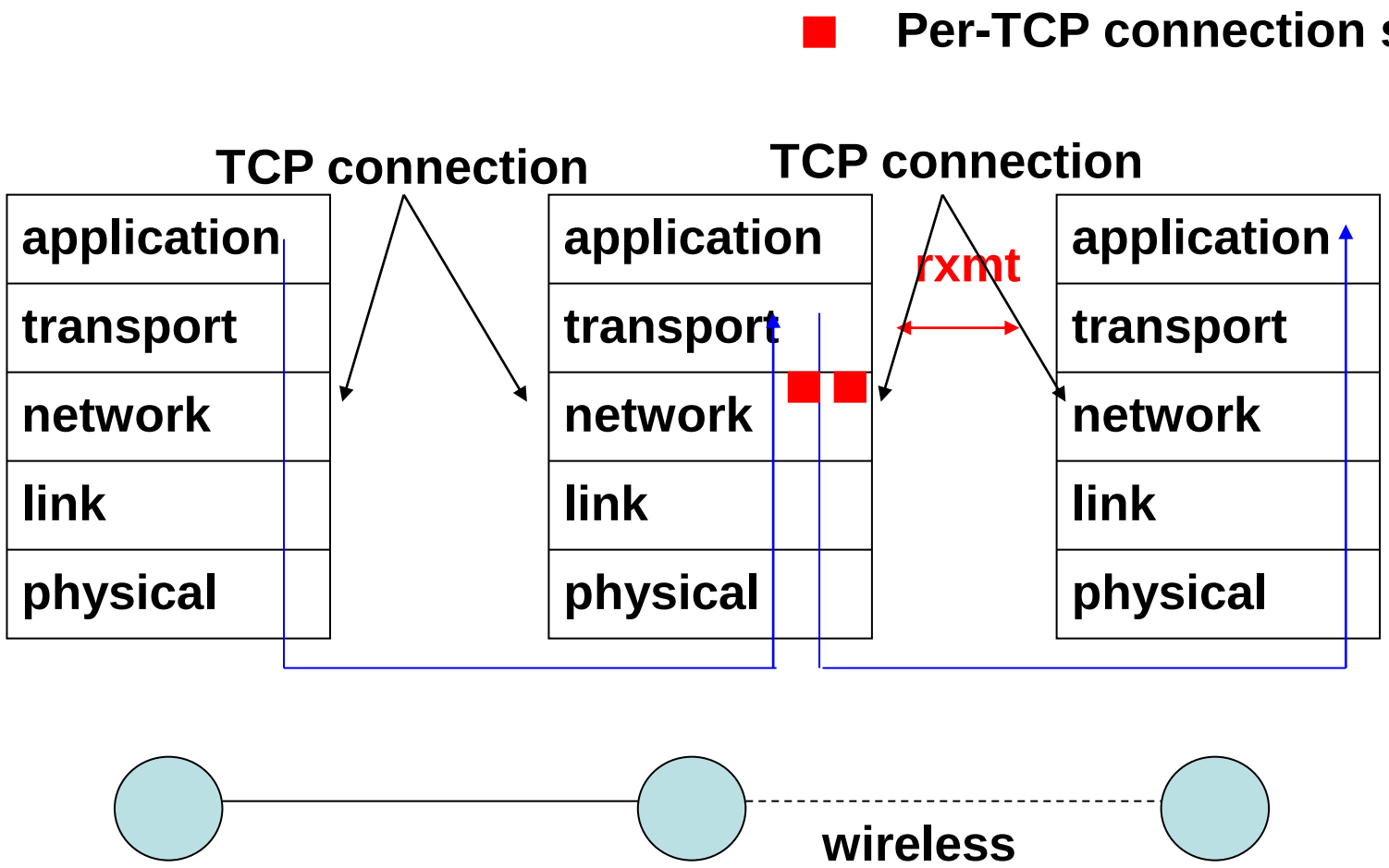
- How many times to retransmit at the link level before giving up?
- What triggers link level retransmissions?
- How much time is required for a link layer retransmission?
- Should the link layer deliver packets as they arrive, or deliver them in-order?

Split connection approach

- End-to-end TCP connection is broken into one connection on the wired part of route and one over wireless part of the route
- FH-MH = FH-BS + BS-MH



I-TCP: Split connection



Source: Vaidya

I-TCP advantages

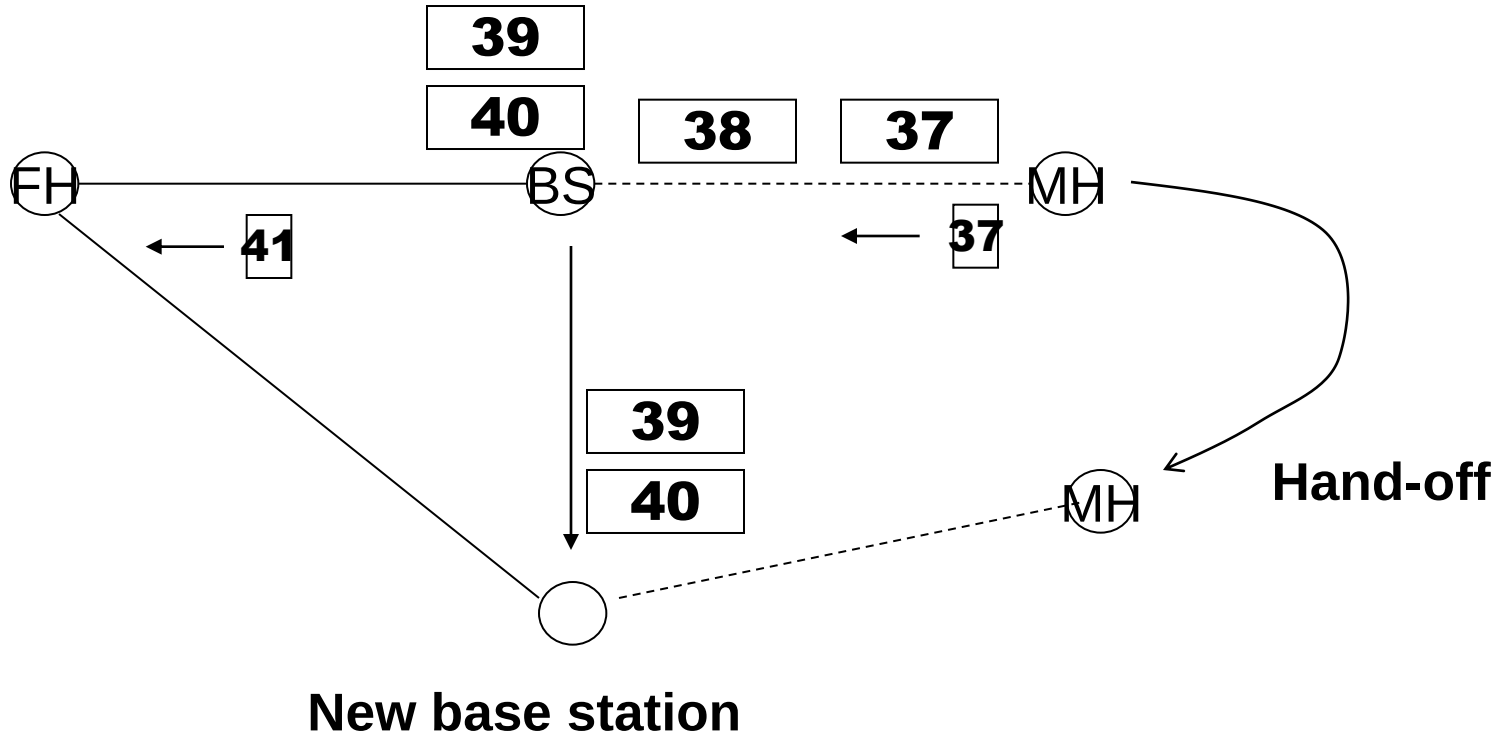
- No changes to TCP for FH
- BS-MH connection can be optimized independent of FH-BS connection
  - Different flow / error control on the two connections
  - Faster recovery due to relatively shorter RTT on wireless link

I-TCP disadvantages

- End-to-end semantics violated
  - ack may be delivered to sender, before data delivered to the receiver
- BS retains hard state
  - Buffer space required at BS on a per-TCP-connection basis
  - BS failure can result in permanent loss of data (unreliability)
  - Hand-off latency increases

Hand-off in I-TCP

- Data that has been ack'd to sender, must be moved to new base station



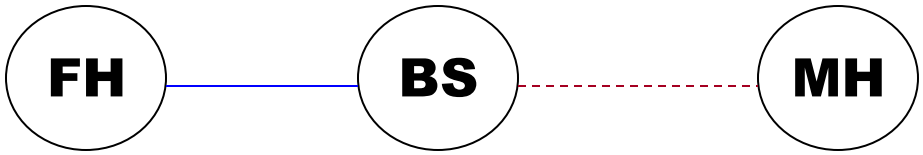
Snoop Protocol

- Retains local recovery of Split Connection approach and uses link level retransmission

- Improves on split connection
  - end-to-end semantics retained
  - soft state at base station, instead of hard state

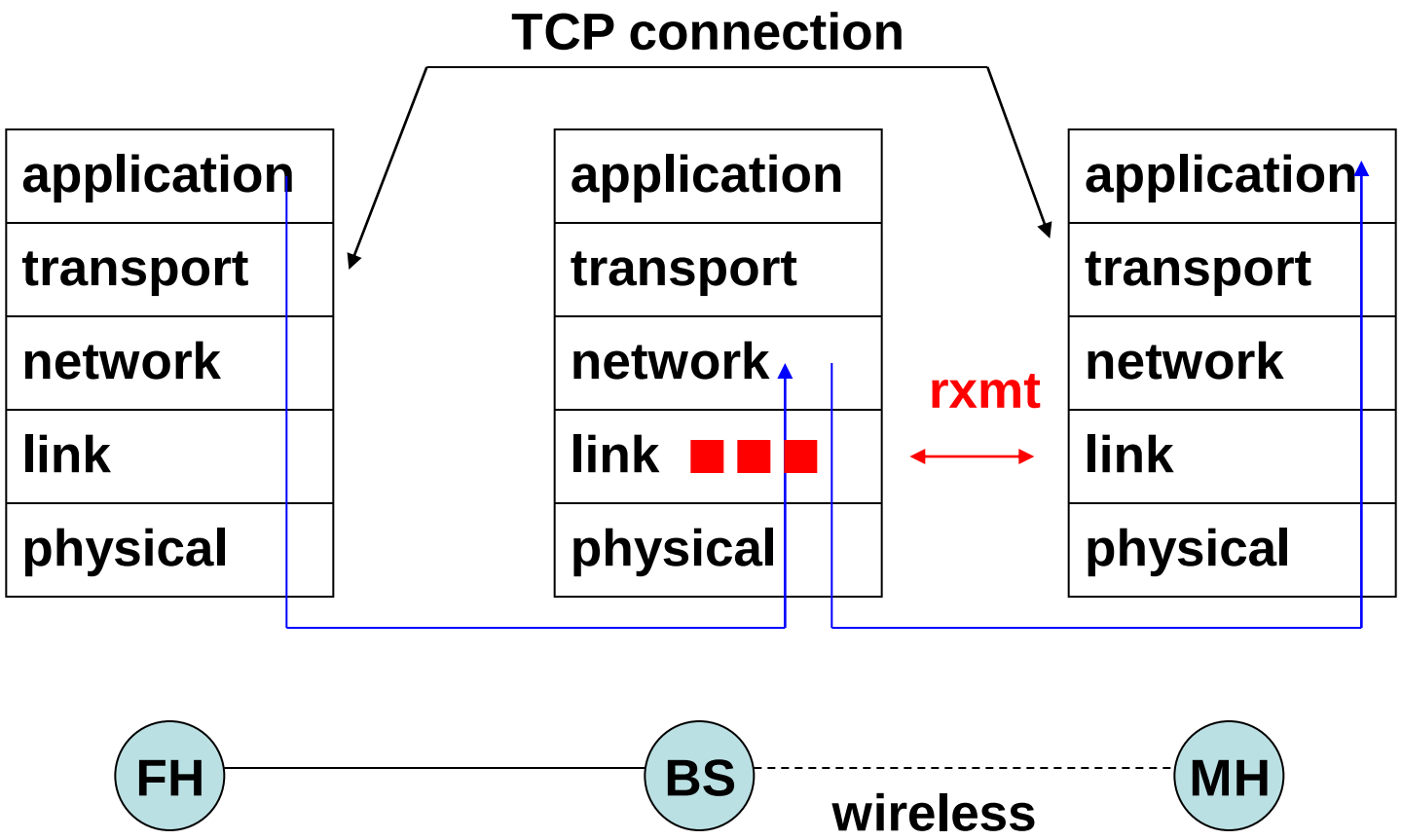
Snoop Protocol

- Buffers data packets at the base station BS
  - to allow link layer retransmission
- When duplicate ACK received by BS from MH
  - retransmit on wireless link, if packet present in buffer
  - drop duplicate ACK
- Prevents fast retransmit at TCP sender FH



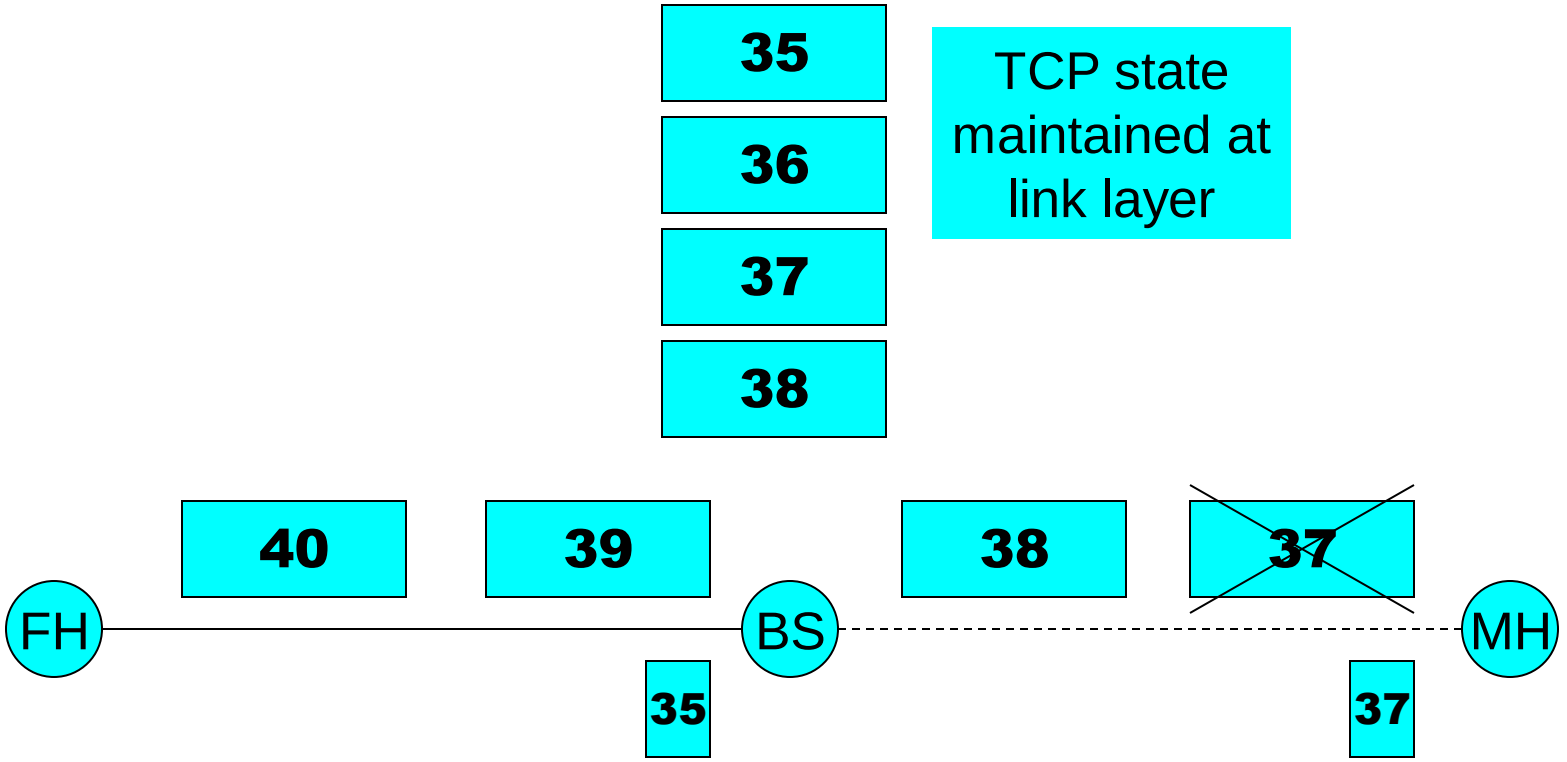
Snoop Protocol

- Per TCP-connection state

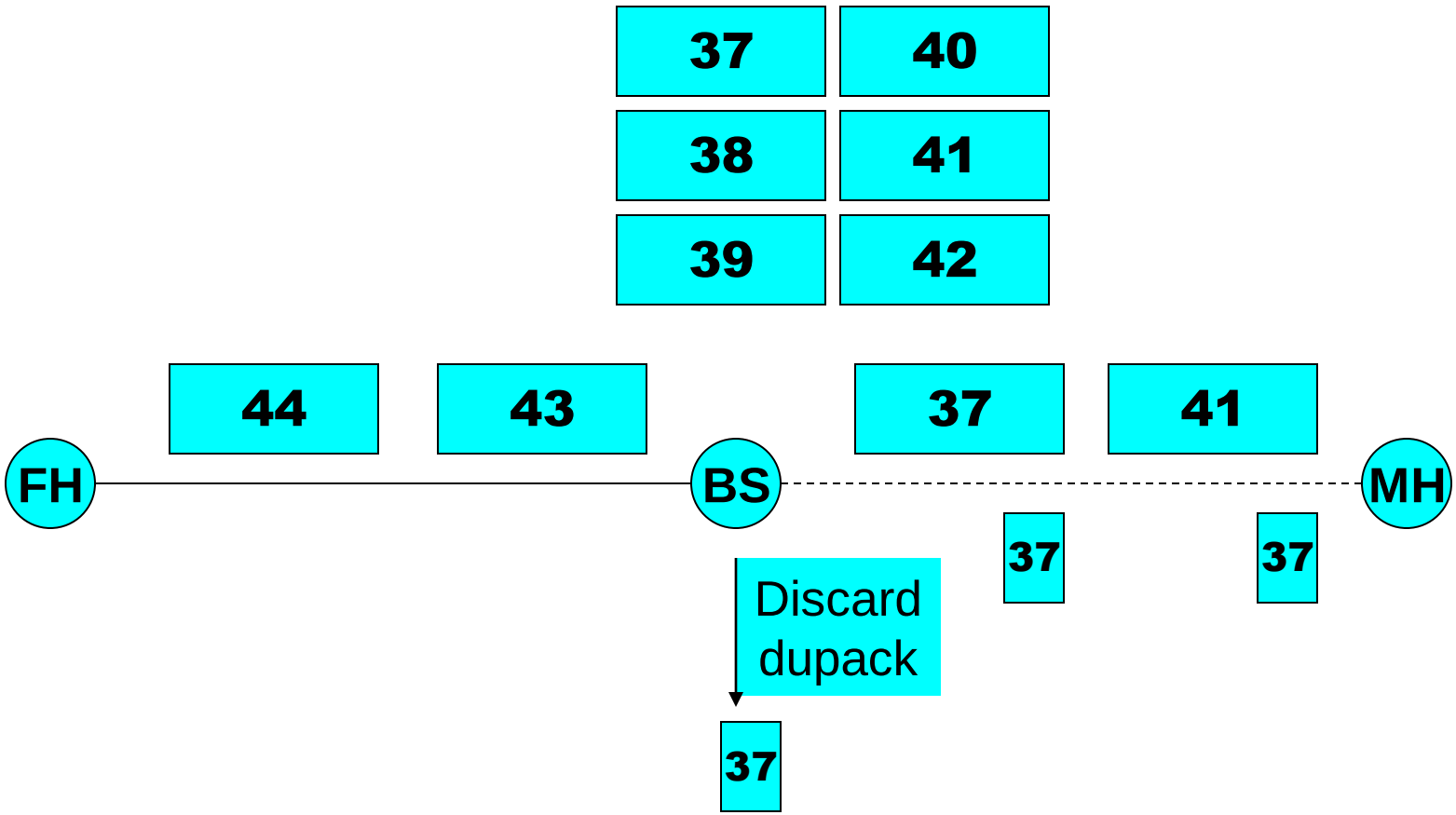


Source: Vaidya

Snoop : Example



Snoop : Example



Snoop advantages

- Local recovery from wireless losses
- Fast retransmit not triggered at sender despite out-of-order link layer delivery
- High throughput can be achieved
- End-to-end semantics retained
- Soft state at base station
  - loss of the soft state affects performance, but not correctness

Snoop disadvantages

- Link layer at base station needs to be TCP-aware
- Not useful if TCP headers are encrypted (IPsec)

Delayed Dupacks

- Attempts to imitate Snoop, without making the base station TCP-aware

- Delayed Dupacks implements the same two features
  - at BS : link layer retransmission
  - at MH : reducing interference between TCP and link layer retransmissions (by delaying dupacks)

Delayed Dupacks

- TCP receiver delays dupacks for interval D, when out-of-order packets received
  - Dupack delay intended to give link level retransmit time to succeed

- Benefit: can result in recovery from a transmission loss without triggering a response from the TCP sender

Delayed dupacks advantages

- Link layer need not be TCP-aware
- Can be used even if TCP headers are encrypted
- Works well for relatively small wireless RTT (compared to end-to-end RTT)
  - relatively small delay D sufficient in such cases

Delayed dupacks disadvantages

- Right value of dupack delay D dependent on the wireless link properties
- Mechanisms to automatically choose D needed
- Delays dupacks for congestion losses too, delaying congestion loss recovery

Mobility and handoff

- Hand-offs may result in temporary loss of route to MH
  - with non-overlapping cells, it may be a while before the mobile host receives a beacon from the new BS

- While routes are being reestablished during handoff, MH and old BS may attempt to send packets to each other, resulting in loss of packets

Impact of handoff

- Split connection approach
  - hard state at base station must be moved to new base station
- Snoop protocol
  - soft state need not be moved
  - while the new base station builds new state, packet losses may not be recovered locally

Handoff issues

- During the long delay for a handoff to complete
  - a whole window worth of data may be lost
- After handoff is complete
  - acks are not received by the TCP sender

- Sender eventually times out, and retransmits
  - If handoff still not complete, another timeout will occur
- Performance penalty
  - Time wasted until timeout occurs
  - Window shrunk after timeout

Using Fast Retransmit

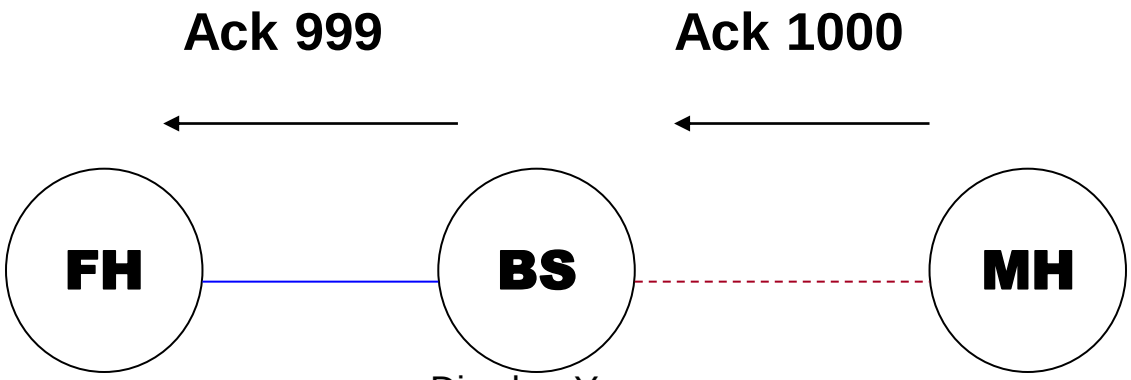
- When MH is the TCP receiver:
  - after handoff is complete, it sends 3 dupacks to the sender
  - this triggers fast retransmit at the sender
- When MH is the TCP sender:
  - invoke fast retransmit after completion of handoff

Mobile TCP (M-TCP)

- Handling of lengthy or frequent disconnections
- M-TCP splits as I-TCP does
  - unmodified TCP for FH to BS
  - optimized TCP for BS to MH
- BS (Foreign Agent)
  - monitors all packets, if disconnection detected
    - set advertised window size to 0
    - sender automatically goes into persistent mode
  - no caching, no retransmission at the BS
    - If a packet is lost on the wireless link, it has to be retransmitted by the original sender

M-TCP

- BS does not send an ack to FH, unless BS has received an ack from MH
  - maintains end-to-end semantics
- BS withholds ack for the last byte ack'd by MH
- When BS does not receive ACK for sometime, it chokes sender by setting advertise window to 0



M-TCP

- When a new ack is received with receiver's advertised window = 0, the sender enters persist mode
- Sender does not send any data in persist mode
  - except when persist timer goes off

- When a positive window advertisement is received, sender exits persist mode
- On exiting persist mode, RTO and cwnd are same as before the persist mode

M-TCP

- Avoids reduction of congestion window due to handoff, unlike the fast retransmit scheme

- Is not reducing the window a good idea?
  - When host moves, route changes, and new route may be more congested
  - It is not obvious that starting full window after handoff is right

FreezeTCP

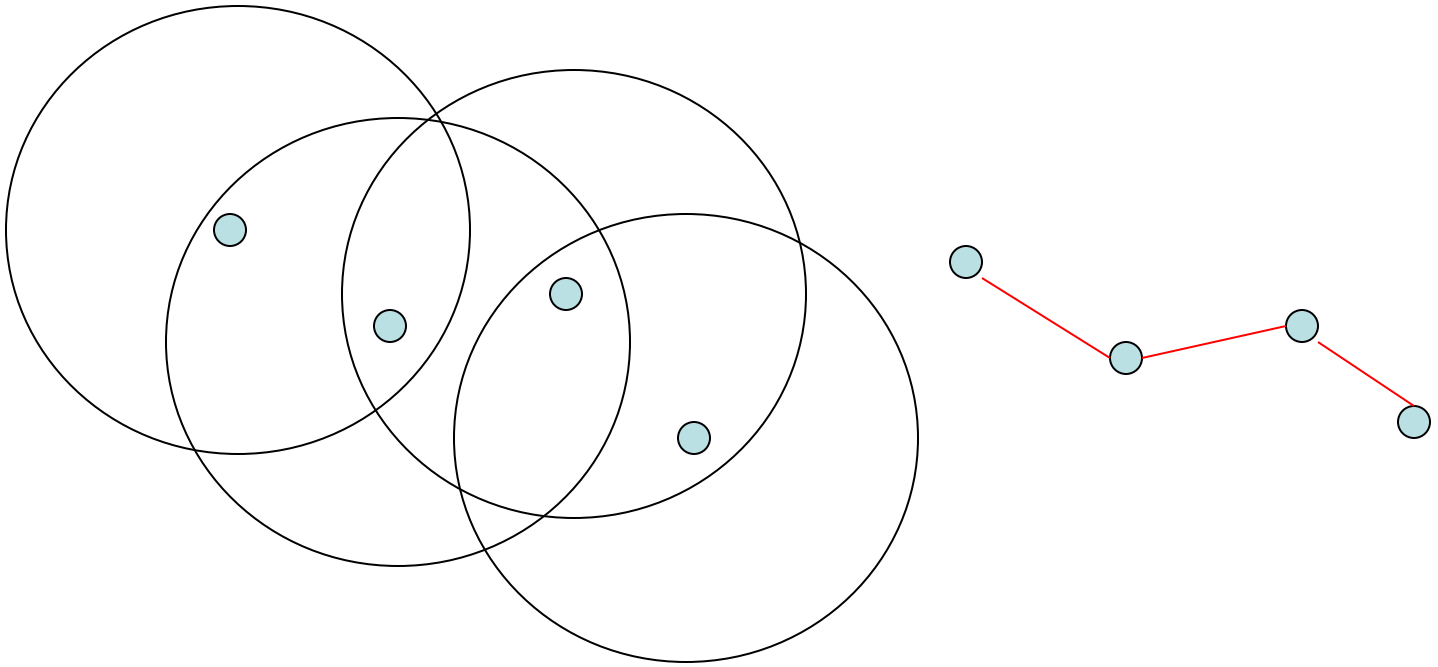
- M-TCP needs help from base station (BS)
  - BS withholds ack for one byte
  - BS uses this ack to send a zero window advertisement when MH moves to another cell
- **FreezeTCP**
  - Receiver sends zero window advertisement (ZWA), upon impending disconnection
  - Receiver sends full window advertisement (FWA), upon reconnection

FreezeTCP

- TCP receiver determines if a handoff is about to happen
  - determination may be based on signal strength
- Receiver should attempt to send ZWA 1 RTT before handoff
- Receiver sends 3 dupacks when route is reestablished
- No help needed from the base station

Multi-hop Wireless (MANET)

- Mobility causes route changes



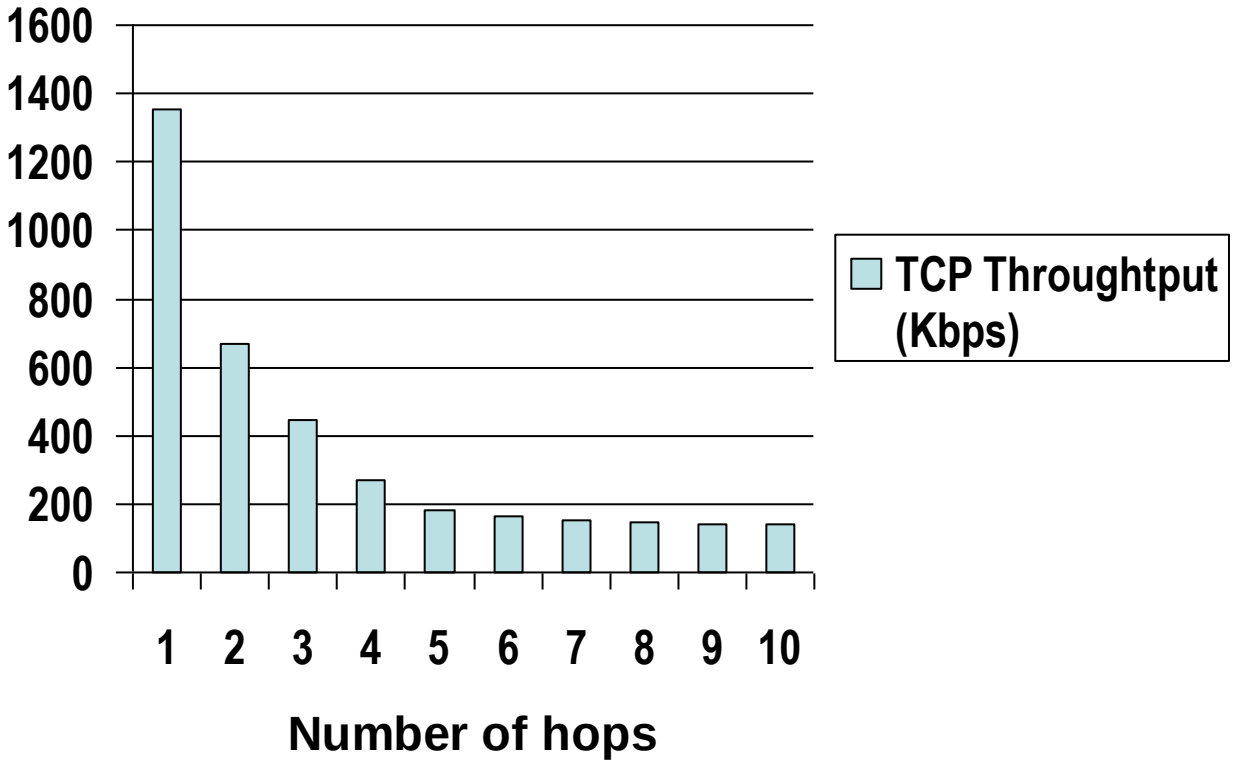
TCP Issues

- Route changes due to mobility
- Wireless transmission errors
  - problem compounded with multiple hops
- Out-of-order packet delivery
  - frequent route changes may cause out-of-order delivery
- Multiple access protocol
  - choice of MAC protocol can impact TCP performance significantly

TCP over multi hop wireless

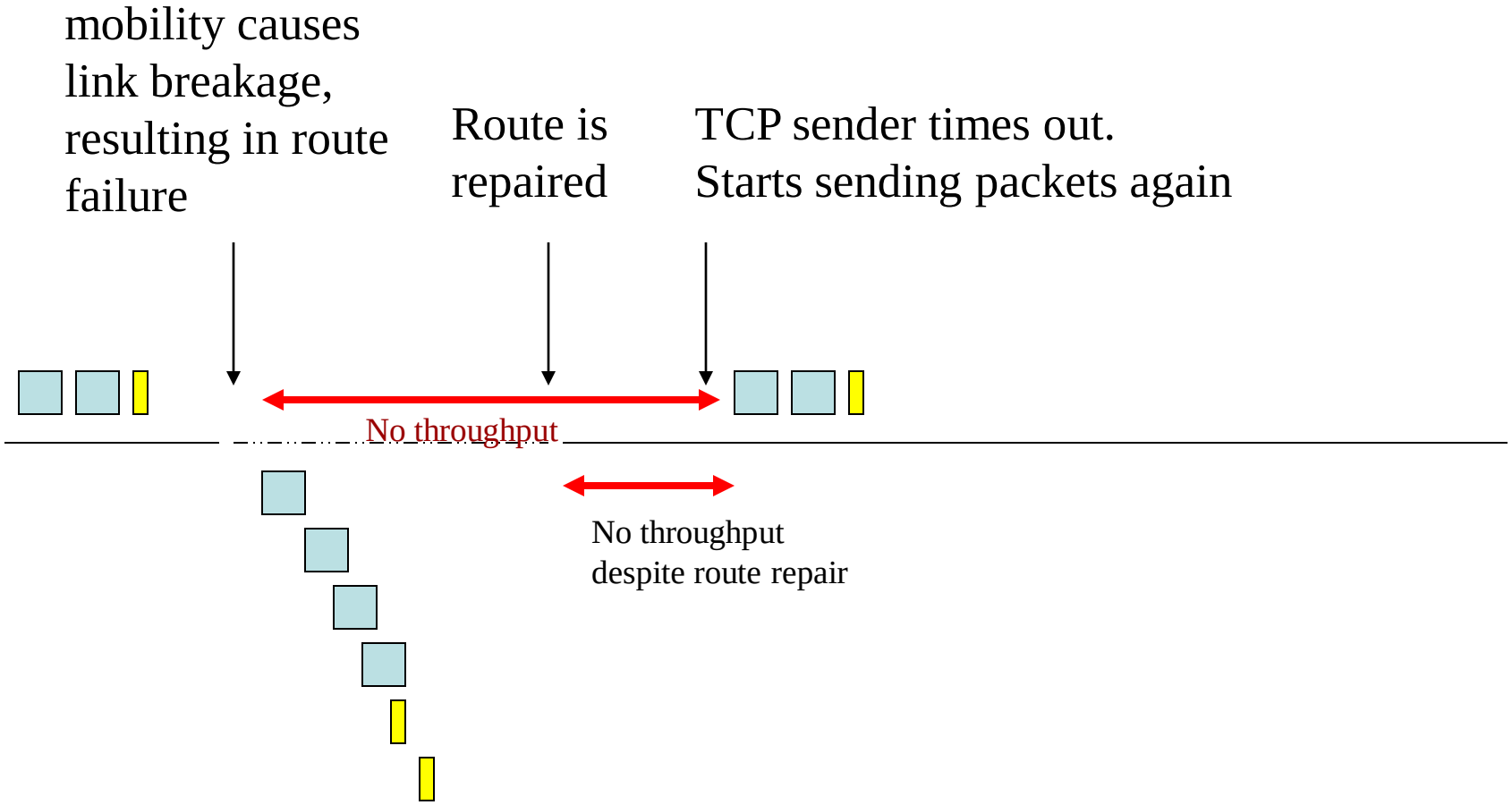
- When contention-based MAC protocol is used, connections over multiple hops are at a disadvantage compared to shorter connections
  - because they have to contend for wireless access at each hop
  - extent of packet delay or drop increases with number of hops

Impact of Multi-Hop Wireless Paths

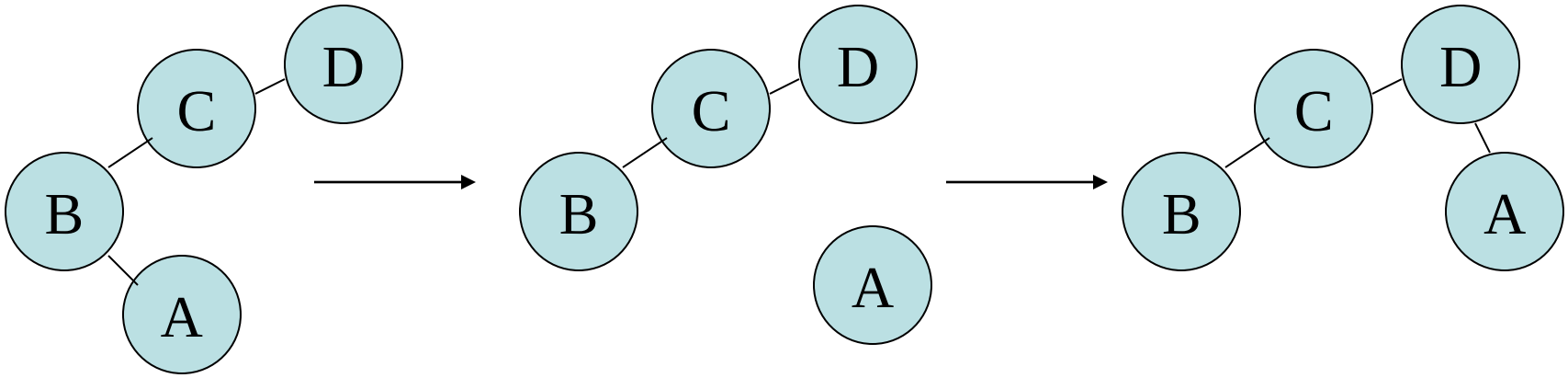


TCP Throughput using 2 Mbps 802.11 MAC

Impact of mobility



Positive impact of mobility



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.

Throughput improves because number of hops reduced.

Improving throughput

- Network feedback
- Inform TCP of route failure by explicit message
- Let TCP know when route is repaired
  - Probing
  - Explicit notification

- Reduces repeated TCP timeouts and backoff

Network Feedback

- Network feedback beneficial
- Need to modify transport & network layer to receive/send feedback
- Need mechanisms for information exchange between layers

References

- **Bakre, A., Badrinath, B., "I-TCP: Indirect TCP for mobile hosts"- IEEE ICDCS 1995.**
- **Balakrishnan, H., Srinivasan, S., Amir, E., and Katz, R., "Improving TCP/IP Performance over Wireless Networks" – ACM Mobicom 1995.**
- **Brown, K., Singh, S., "M-TCP: TCP for mobile cellular networks" – ACM Computer Communication Review, 27 (5), 1997.**

- **Goff, T. Moronski, J. Phatak, D.S. Gupta, V. "Freeze-TCP: a true end-to-end TCP enhancement mechanism for mobile environments**" – IEEE Infocom 2000.