Introduction to Computer Networks

COSC 4377

Lecture 11

Spring 2012

February 22, 2012

Announcements

- HW5 due today
- Exam1 on Monday
  - You can bring one page of notes

Today's Topics

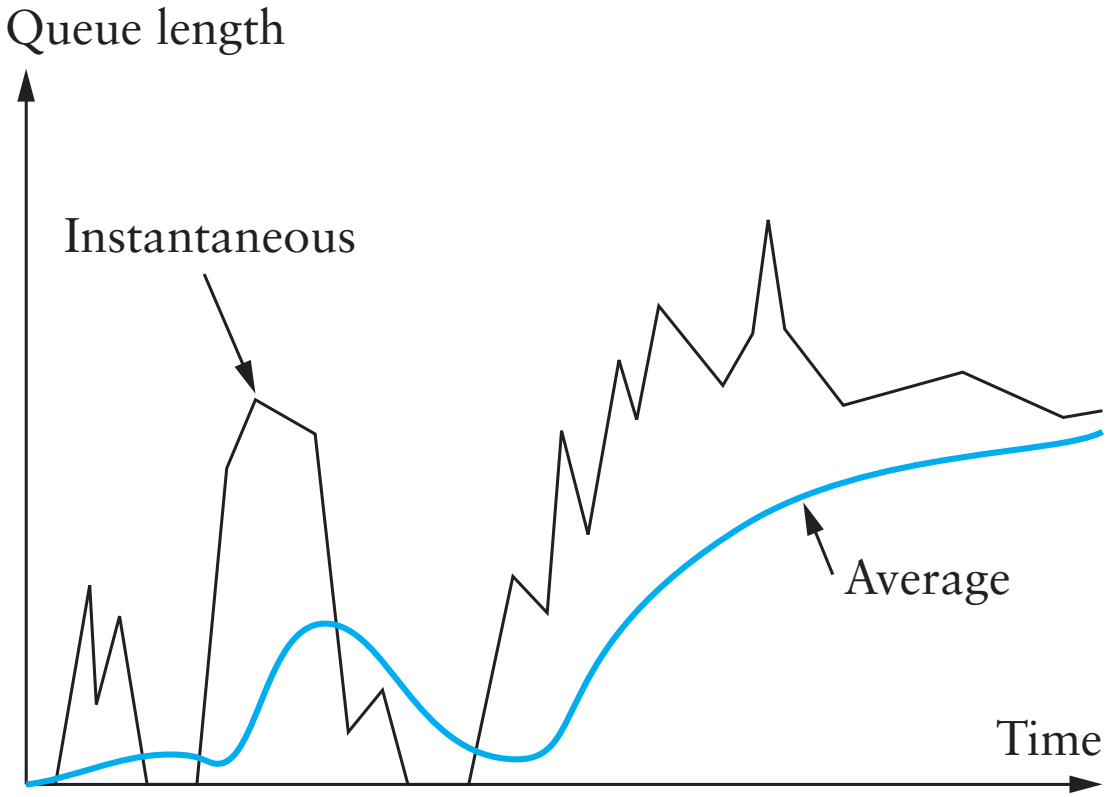
- HW5 discussions
- Transport Protocol
  - TCP Friendliness
  - Getting help from the network

Help from the network

- What if routers could *tell* TCP that congestion is happening?
  - Congestion causes queues to grow: rate mismatch
- TCP responds to drops
- Idea: Random Early Drop (RED)
  - Rather than wait for queue to become full, drop packet with some probability that increases with queue length
  - TCP will react by reducing cwnd
  - Could also mark instead of dropping: ECN

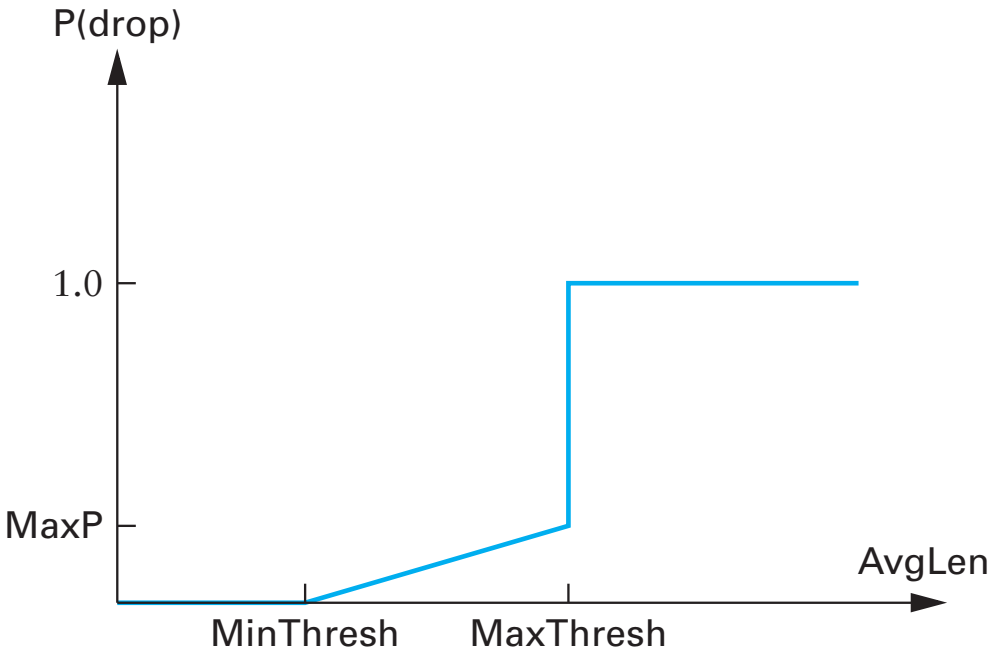
RED Details

- Compute average queue length (EWMA)
  - Don't want to react to very quick fluctuations



RED Drop Probability

- Define two thresholds: MinThresh, MaxThresh
- Drop probability:



- **Improvements to spread drops**

RED Advantages

- Probability of dropping a packet of a particular flow is roughly proportional to the share of the bandwidth that flow is currently getting
- Higher network utilization with low delays
- Average queue length small, but can absorb bursts
- ECN
  - Similar to RED, but router sets bit in the packet
  - Must be supported by both ends
  - Avoids retransmissions optionally dropped packets

More help from the network

- Problem: still vulnerable to malicious flows!
  - RED will drop packets from large flows preferentially, but they don't have to respond appropriately
- Idea: Multiple Queues (one per flow)
  - Serve queues in Round-Robin
  - Nagle (1987)
  - Good: protects against misbehaving flows
  - Disadvantage?
  - Flows with larger packets get higher bandwidth

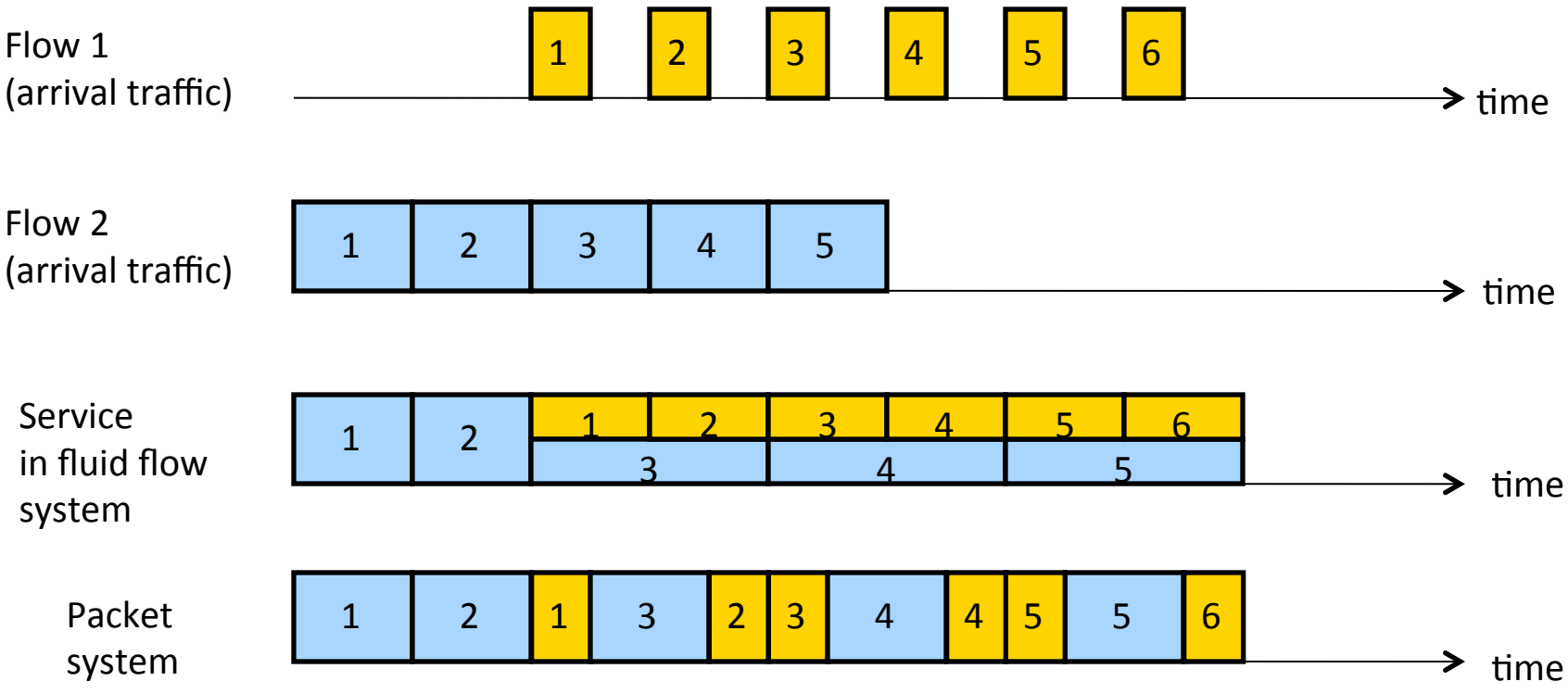
Solution

- Bit-by-bit round robing
- Can we do this?
  - No, packets cannot be preempted!
- We can only approximate it…

Fair Queueing

- Define a *fluid flow* system as one where flows are served bit-by-bit
- Simulate *ff*, and serve packets in the order in which they would finish in the *ff* system
- Each flow will receive exactly its fair share

Example


Flow 1
(arrival traffic)
1
2
3
4
5
6
time
Flow 2
(arrival traffic)
1
2
3
4
5
time
Service
in fluid flow
system
1
2
1
2
3
4
5
6
3
4
5
time
Packet
system
1
2
1
3
2
3
4
4
5
5
6
time

Implementing FQ

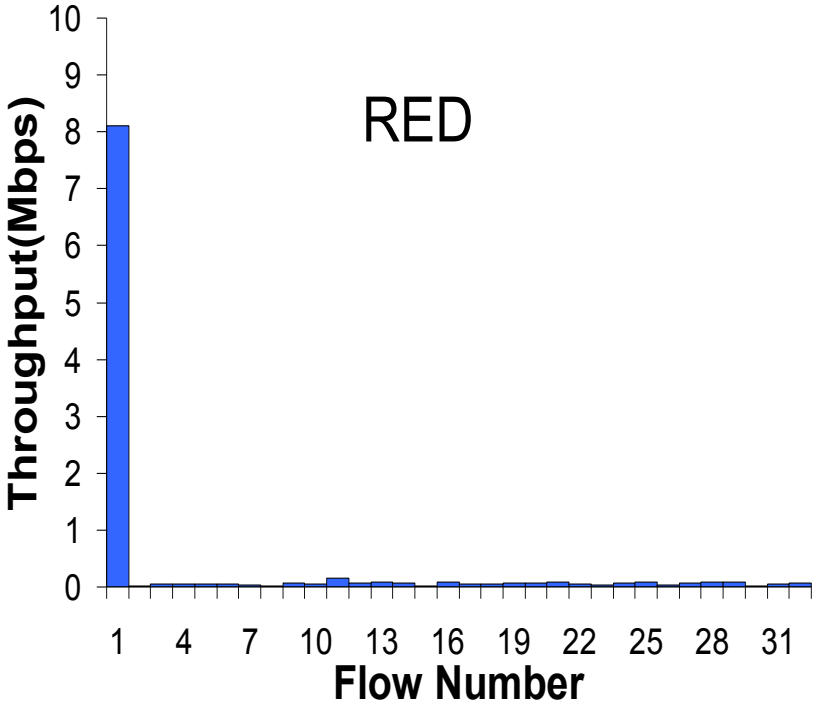
- Suppose clock ticks with each bit transmitted
  - (RR, among all active flows)
- P_i is the length of the packet
- S_i is packet i's start of transmission time
- F_i is packet i's end of transmission time
- $F_i = S_i + P_i$

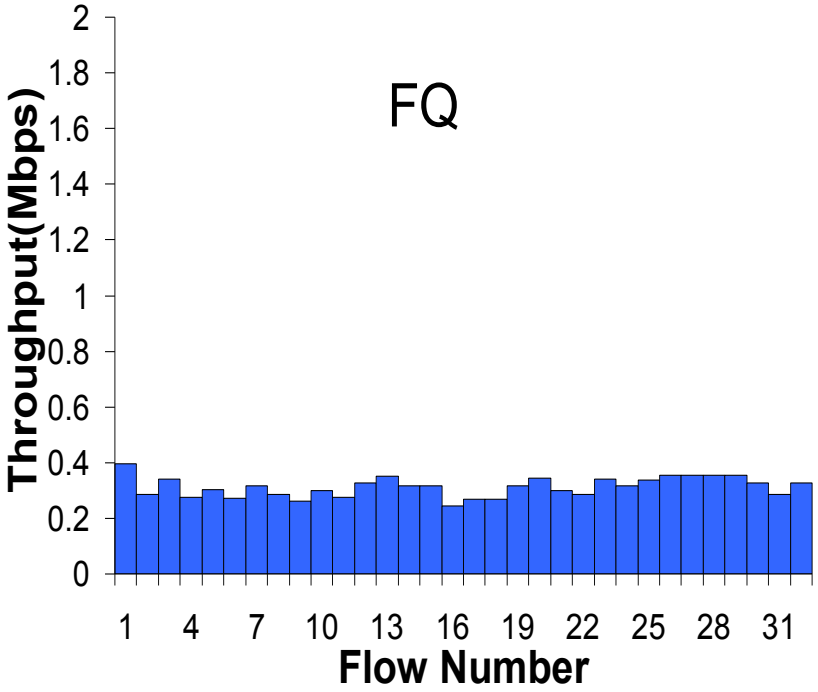
Fair Queueing

- Across all flows
  - Calculate F_i for each packet that arrives on each flow
  - Next packet to transmit is that with the lowest F_i
  - Clock rate depends on the number of flows
- Advantages
  - Achieves max-min fairness, independent of sources
  - Work conserving
- Disadvantages
  - Requires non-trivial support from routers
  - Requires reliable identification of flows
  - Not perfect: can't preempt packets

Fair Queueing Example

- 10Mbps link, 1 10Mbps UDP, 31 TCPs





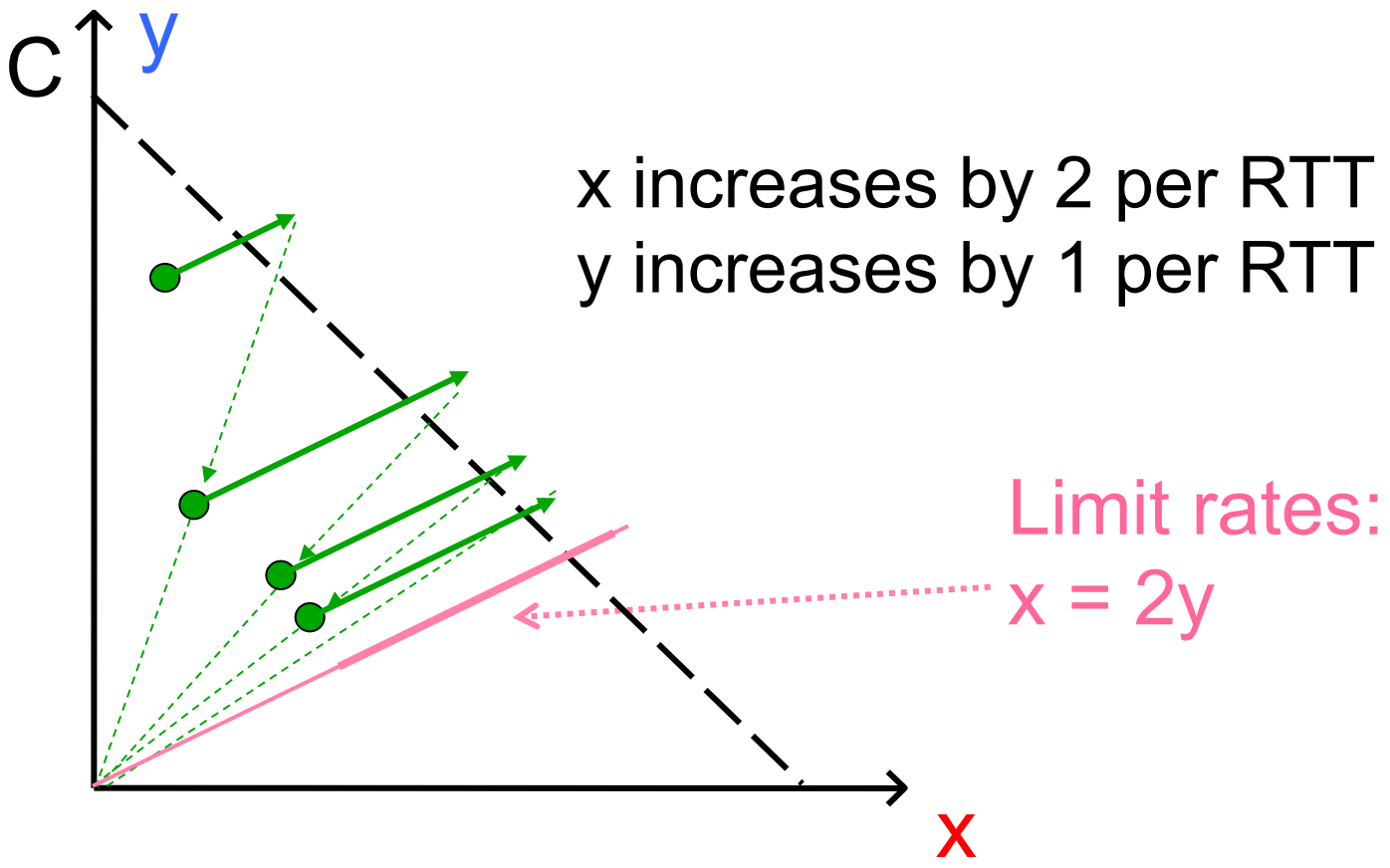
Big Picture

- Fair Queuing doesn't eliminate congestion: just manages it
- You need both, ideally:
  - End-host congestion control to adapt
  - Router congestion control to provide isolation

Cheating TCP

- Three possible ways to cheat
  - Increasing cwnd faster
  - Large initial cwnd
  - Opening many connections
  - Ack Division Attack

Increasing cwnd Faster



x increases by 2 per RTT
y increases by 1 per RTT

Limit rates:
x = 2y

Figure from Walrand, Berkeley EECS 122, 2003

Larger Initial Window

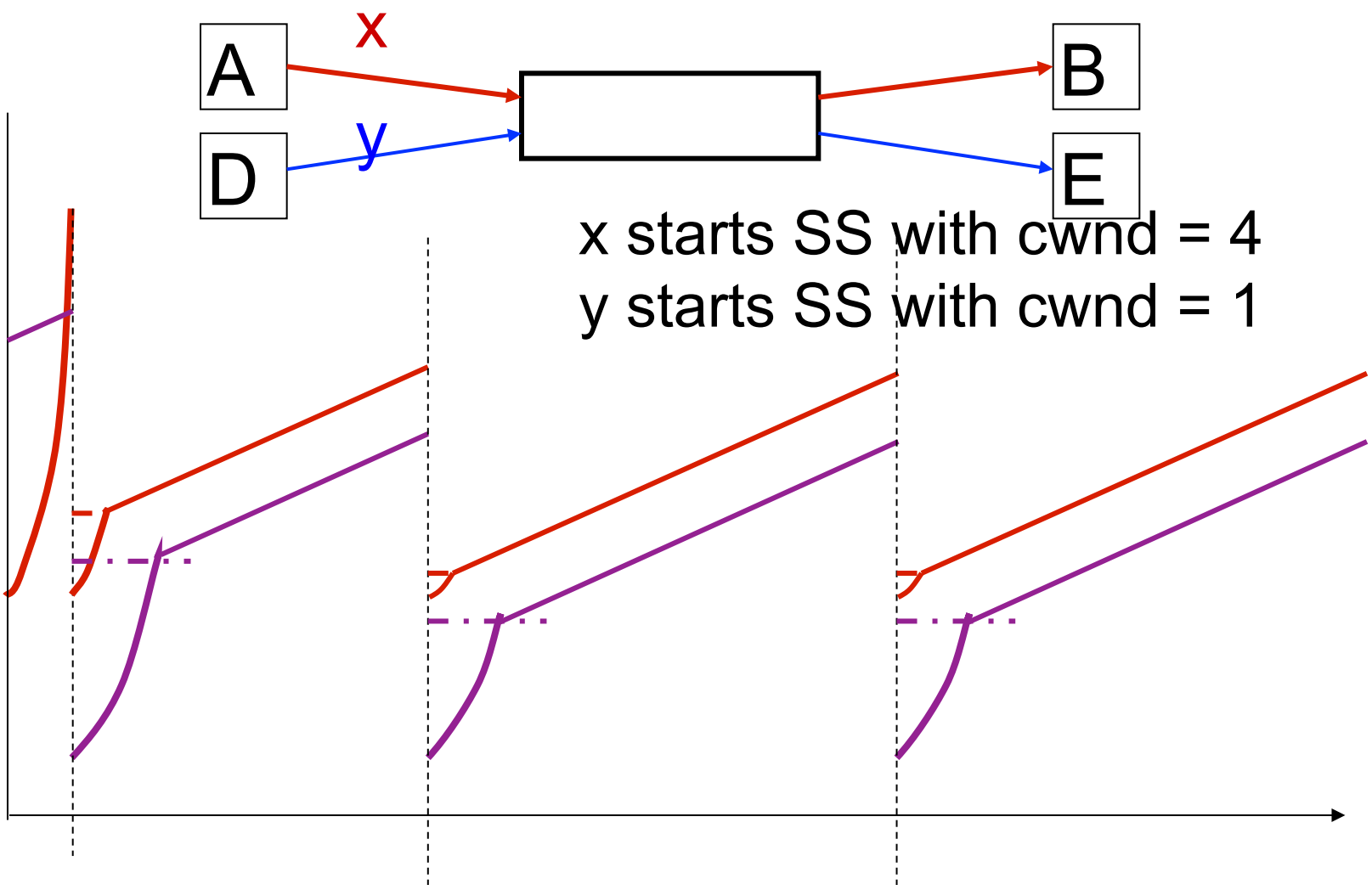
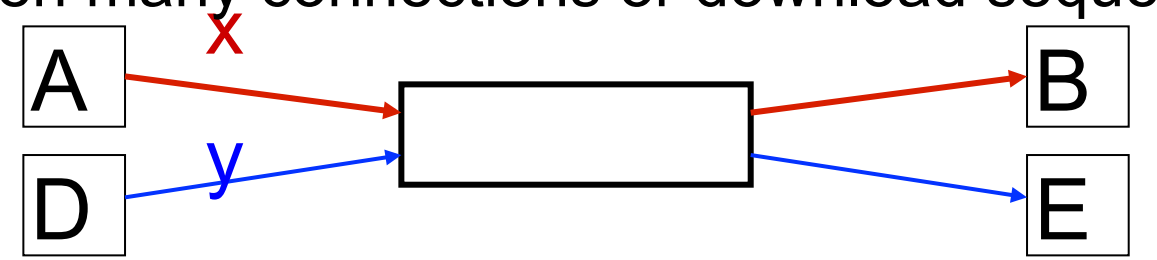


Figure from Walrand, Berkeley EECS 122, 2003

Open Many Connections

- **Web Browser: has to download k objects for a page**
  - Open many connections or download sequentially?



  - Assume:
    - A opens 10 connections to B
    - B opens 1 connection to E
  - TCP is fair among connections
    - A gets 10 times more bandwidth than B

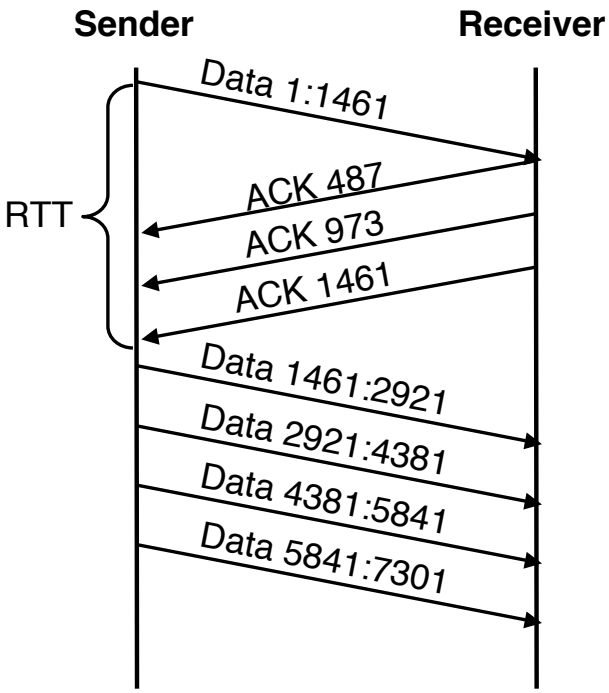
Figure from Walrand, Berkeley EECS 122, 2003

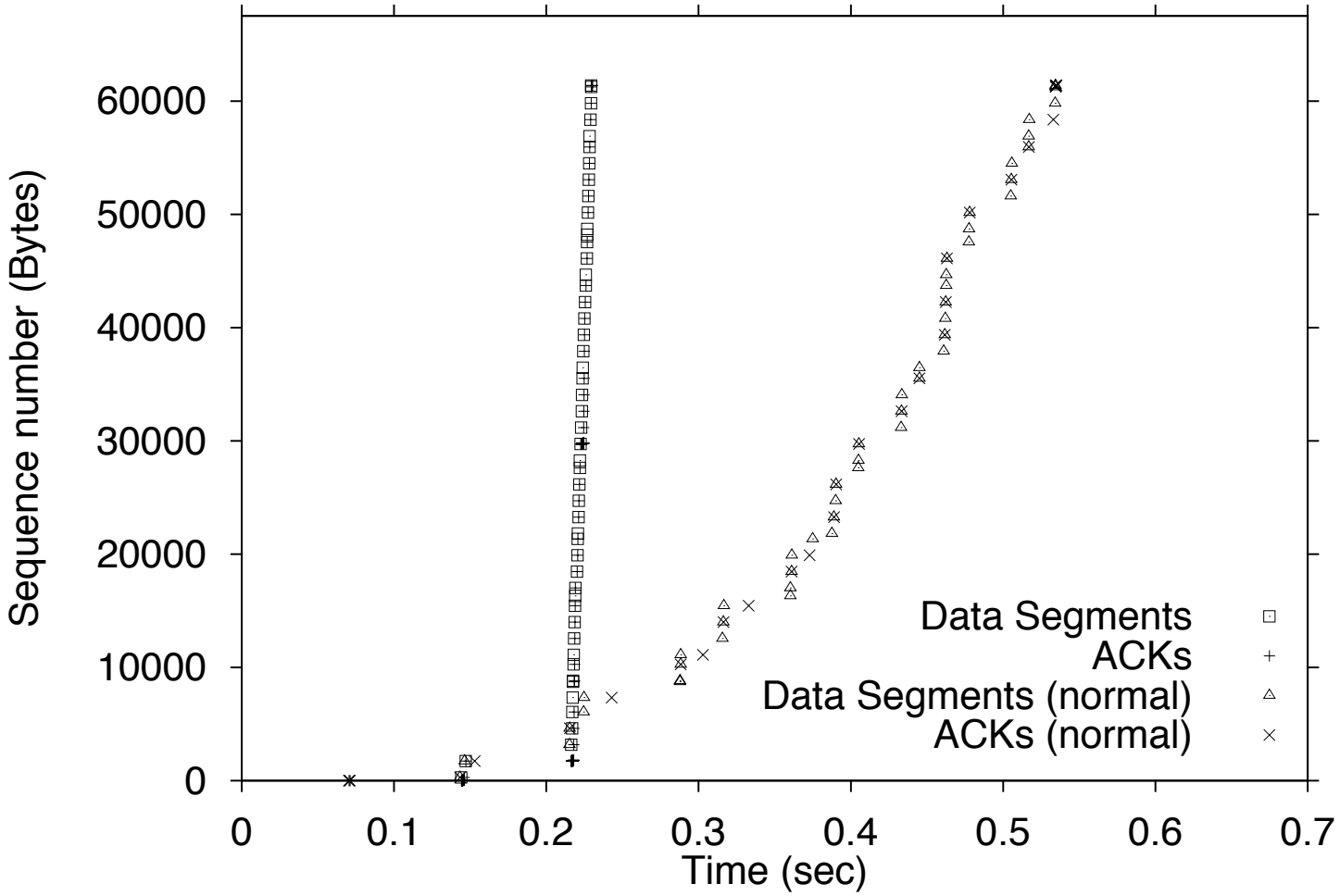
Exploiting Implicit Assumptions

- Savage, et al., CCR 1999:
  - "TCP Congestion Control with a Misbehaving Receiver"
- Exploits ambiguity in meaning of ACK
  - ACKs can specify any byte range for error control
  - Congestion control assumes ACKs cover entire sent segments

ACK Division Attack

- Receiver: "*upon receiving a segment with N bytes, divide the bytes in M groups and acknowledge each group separately*"
- Sender will grow window M times faster
- Could cause growth to 4GB in 4 RTTs!
  - M = N = 1460




Sequence number (Bytes)
60000
50000
40000
30000
20000
10000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Time (sec)
Data Segments
ACKs
Data Segments (normal)
ACKs (normal)

[Savage 99]

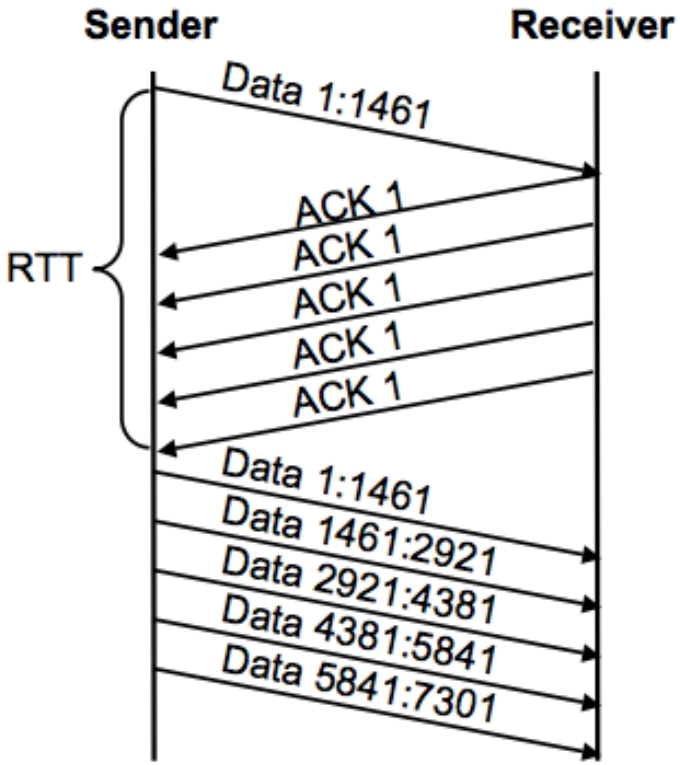
Defense

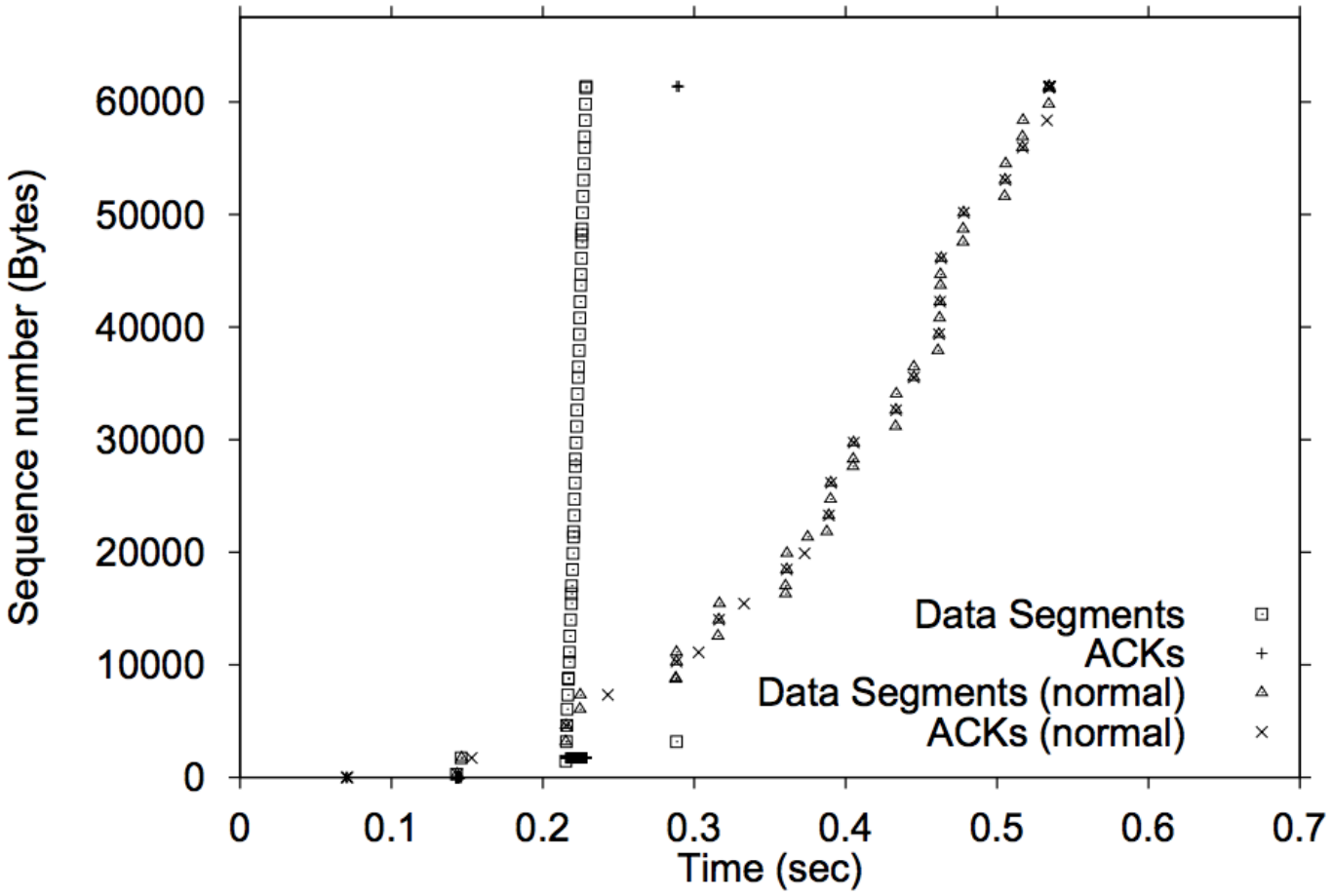
- Appropriate Byte Counting
  - [RFC3465 (2003), RFC 5681 (2009)]
  - In slow start, cwnd += min (N, MSS)

  where N is the number of newly acknowledged bytes in the received ACK

DupACK spoofing

- Receiver: *"Upon receiving a data segment, the receiver sends a long stream of acknowledgments for the last sequence number received"*
- Sender sends at a rate proportional to the ack rate

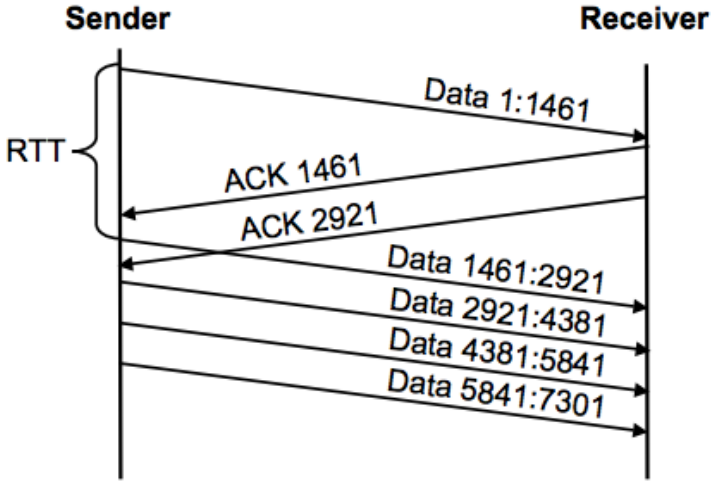


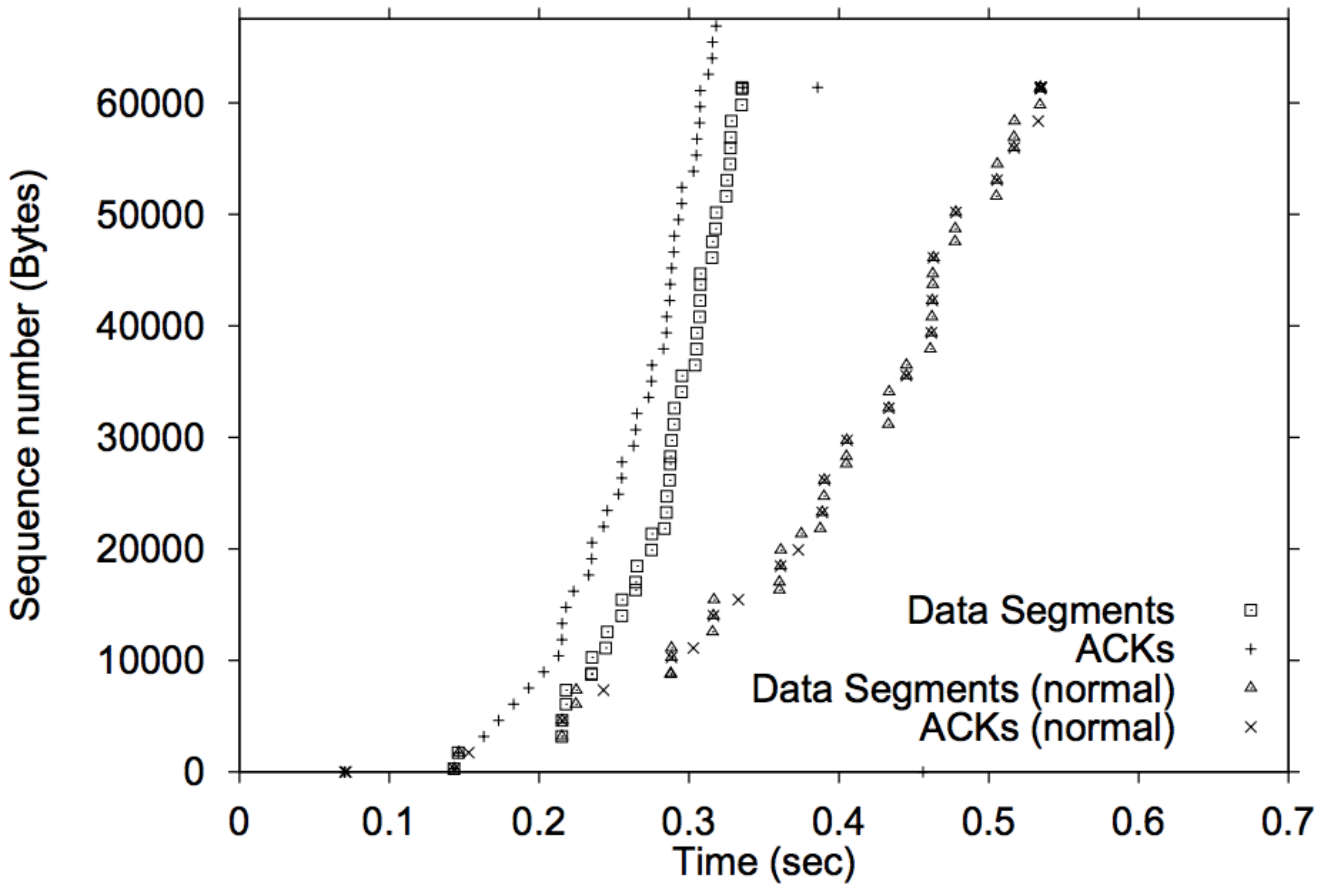

Sequence number (Bytes)
60000
50000
40000
30000
20000
10000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Time (sec)
Data Segments
ACKs
Data Segments (normal)
ACKs (normal)

[Savage 99]

Optimistic ACKing

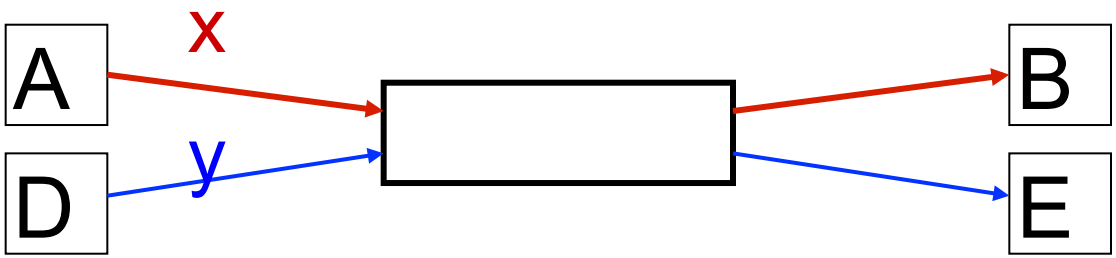
- Receiver: *"Upon receiving a data segment, the receiver sends a stream of acknowledgments anticipating data that will be sent by the sender"*




Sequence number (Bytes)
60000
50000
40000
30000
20000
10000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Time (sec)
Data Segments
ACKs
Data Segments (normal)
ACKs (normal)

[Savage 99]

Cheating TCP and Game Theory



D → Increases by 1 Increases by 5

A ↓

Increases by 1	22, 22	10, 35
Increases by 5	35, 10	15, 15

(x, y)

Too aggressive
→Losses
→Throughput falls

Individual incentives: cheating pays
Social incentives: better off without cheating

Topics for Exam 1

- Characterizing network performance
- Application-layer protocols
  - DNS, HTTP
- Transport protocols

- Miscellaneous topics
  - FTP/HTTP, webpagetest.org, DHCP, Dynamic DNS, SPDY