+

Principles of TCP Performance

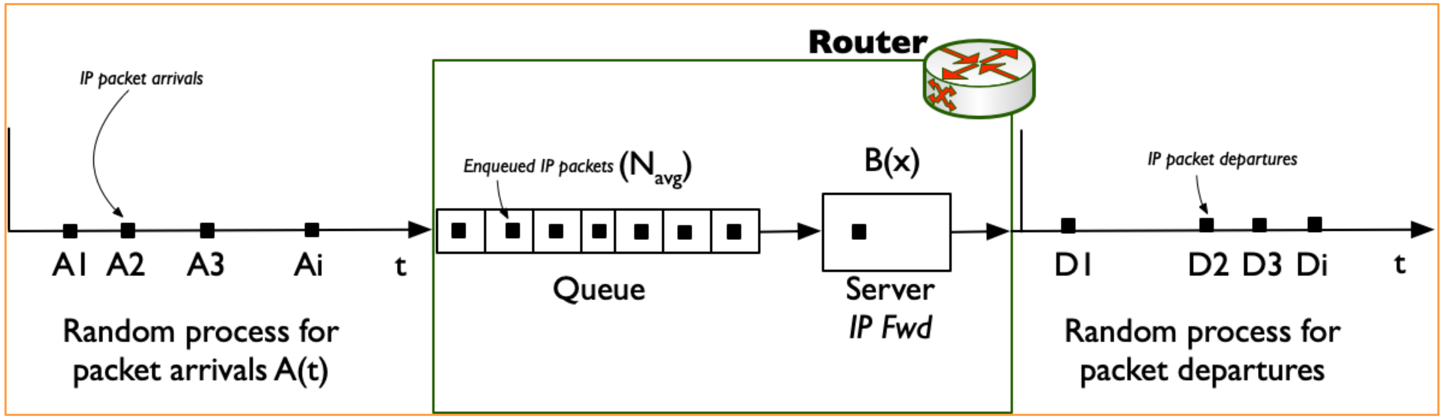
Optimization of TCP end-to-end throughput

Reminder: Routers and switches are all Store and Forward devices

- IP router:
    - Queue for storing incoming packets: **QUEUE**
    - Processor (LPM, IP Fwd alg): **SERVER**

- Input queue is necessary
    - To absorb packets while the server is processing the first of them
    - Otherwise, packets received while server busy, would be dropped

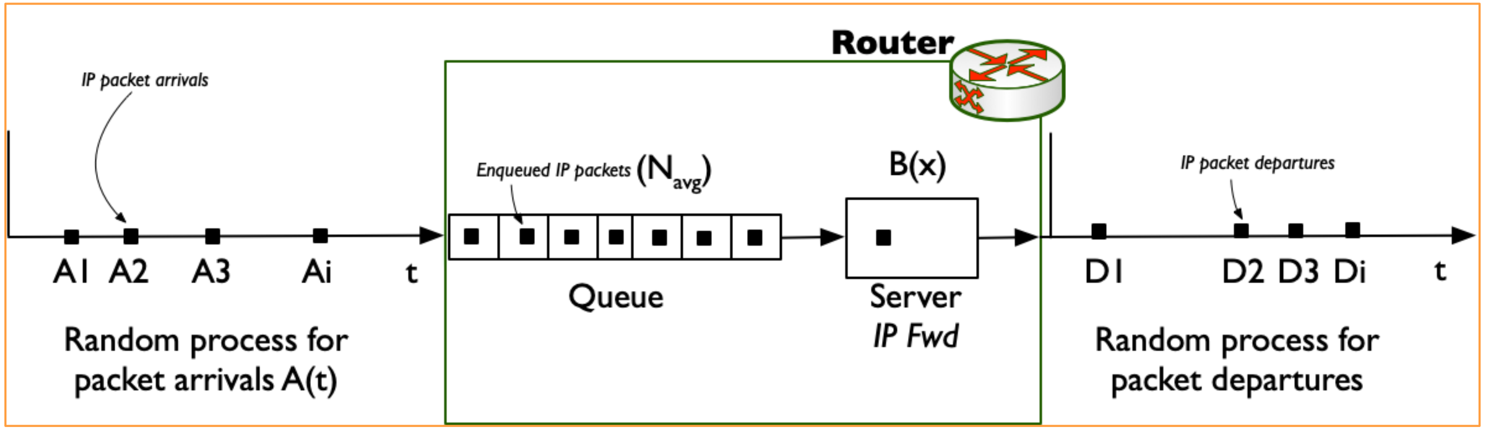
- The time a packet spends waiting in the queue increases the packet's overall delay as it travels to its destination, hopping from one router to the next



Little's Law: Essential for Computer Science

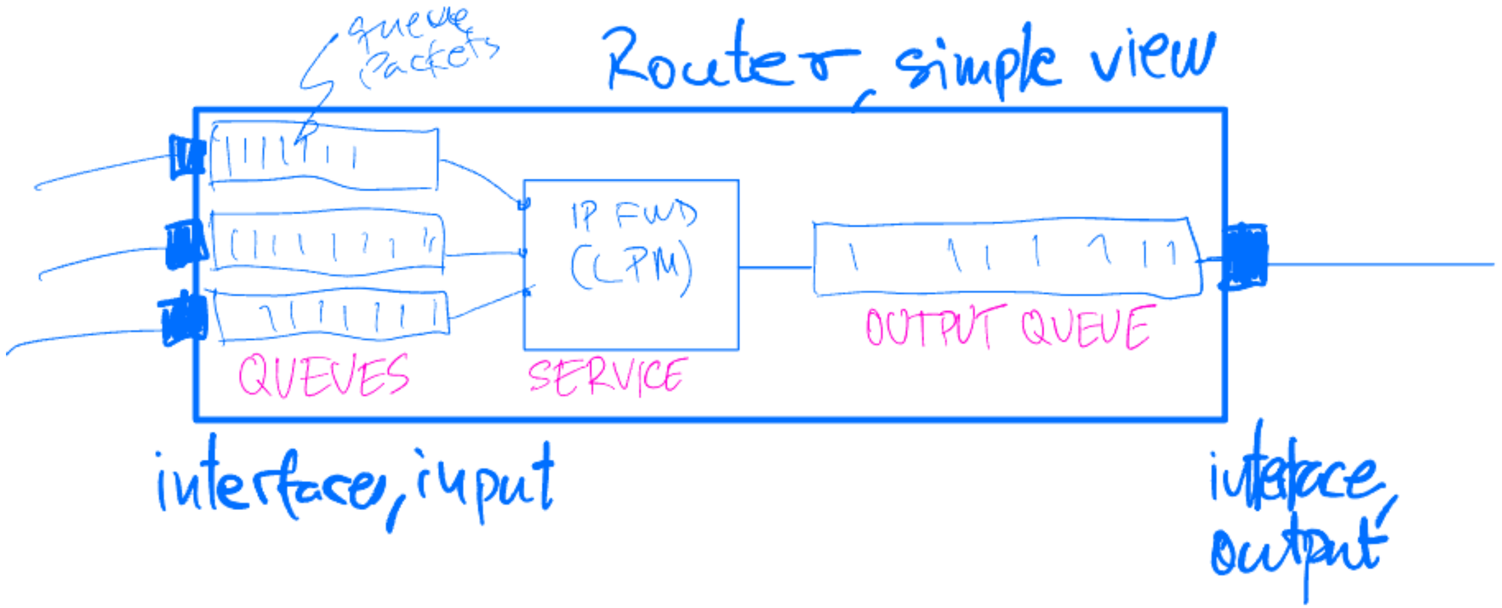
If t is sufficiently large:

- $N_{avg} = \lambda \cdot T_a$
  - The *average* queue length is given by the product of average packet rate and the average residence time
- Little's result is proven to be valid:
  - For all arrival distributions A(t)
  - For all service time distributions B(x)
  - For all queue disciplines (Priority, FIFO, etc)
  - For any number of servers
- **Example.** A(t) is a Poisson distribution, then the interarrivals follow an exponential distribution and the random process is known as Markovian (M). In this case, the probability of receiving a packet in x_{i+1} after receiving a packet in x_i is not affected by the past history of arrivals. This is known as the *memoryless property.* These queues are described by the Kendall's notation as M/M/1: Markovian interarrivals, Markovian Service times, and with 1 server.



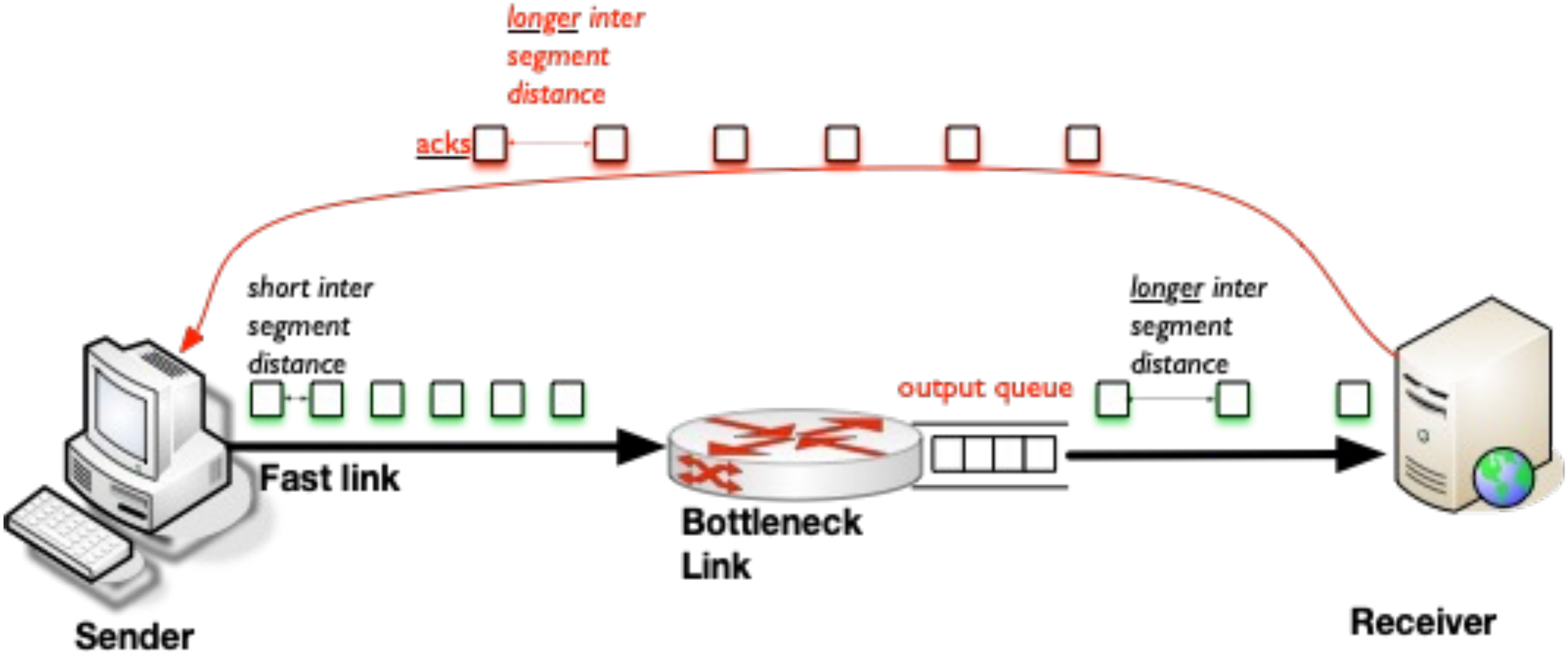
Basic structure of an IP Router

- At this moment, *the output link,* receives traffic from three *input links*
- The output link, when demand is high, queues packets in a buffer
  - Increases the delay undergone by each packet
  - In the limit, when the link is congested, it begins to drop packets (Packets get lost)



Bottleneck link at an IP router

- The bottleneck link limits the maximum number of segments present in the network
- Product 2BD: B is the bandwidth of the bottleneck link and 2D is the Rtt



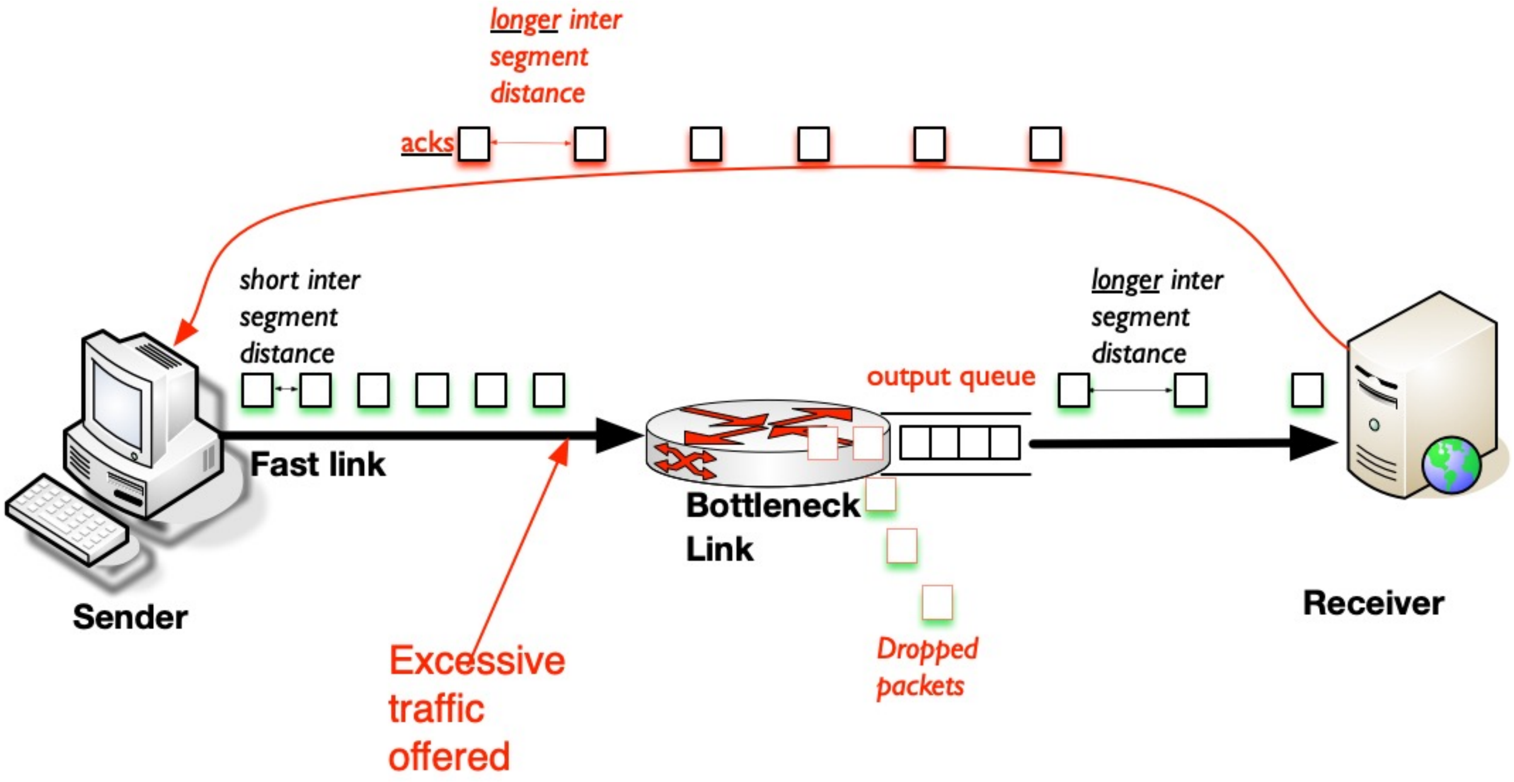
(C) 2016 José María Foces Morán

TCP, congestion control

14-oct-2024

How TCP discovers the end-to-end capacity of a TCP connection

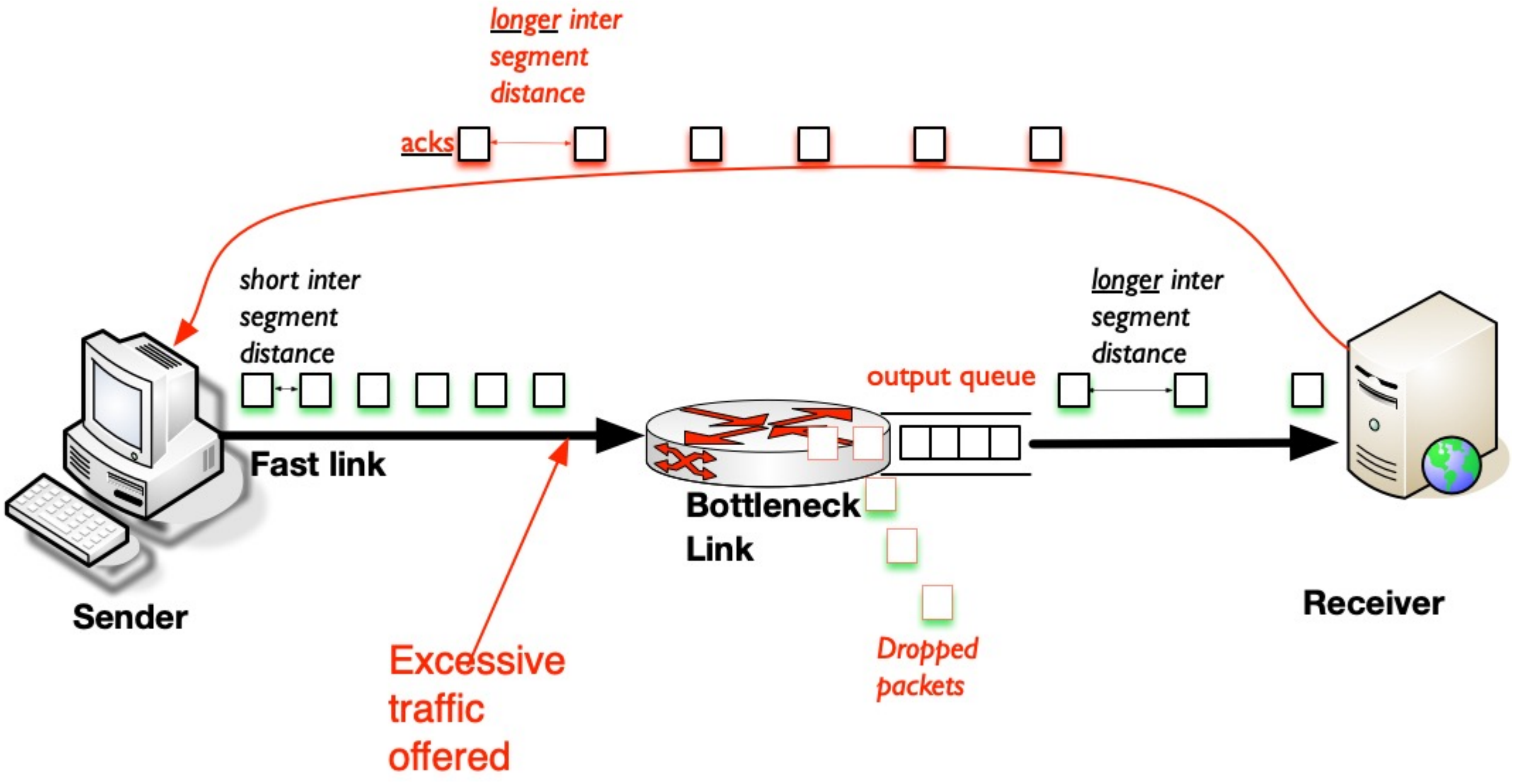
- TCP needs to discover how many packets/sec can be injected into the network, *safely*
- With a limited packet loss



All rights reseved (C) 2020-2024 by José María Foces Morán and José María Foces Vivancos

W_s >= 2BD: The benchmark to TCP

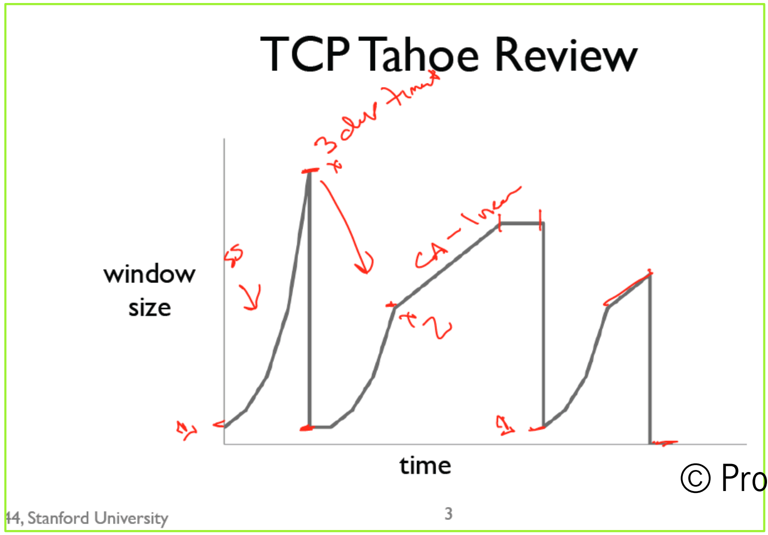
- TCP strives to achieve a Ws that is at least 2BD
  - 2D = Rtt
  - B the bandwidth offered by the bottleneck router which stands on the end-to-end path (C to S)

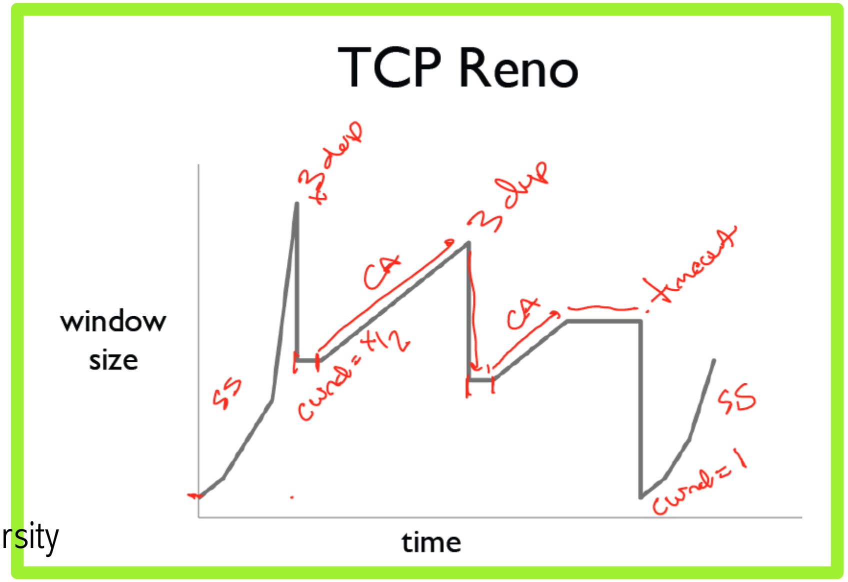


AIMD: Additive Increase, Multiplicative Decrease

14-oct-2024

- TCP needs to discover how many packets/sec can be injected into the network, safely
  - Recall, the benchmark is the end-to-end path's 2BD product
- Without causing packet loss
- The effective TCP's transmit window becomes =MIN (CongestionWindow, AdvWindow)
- CW = CongestionWindow

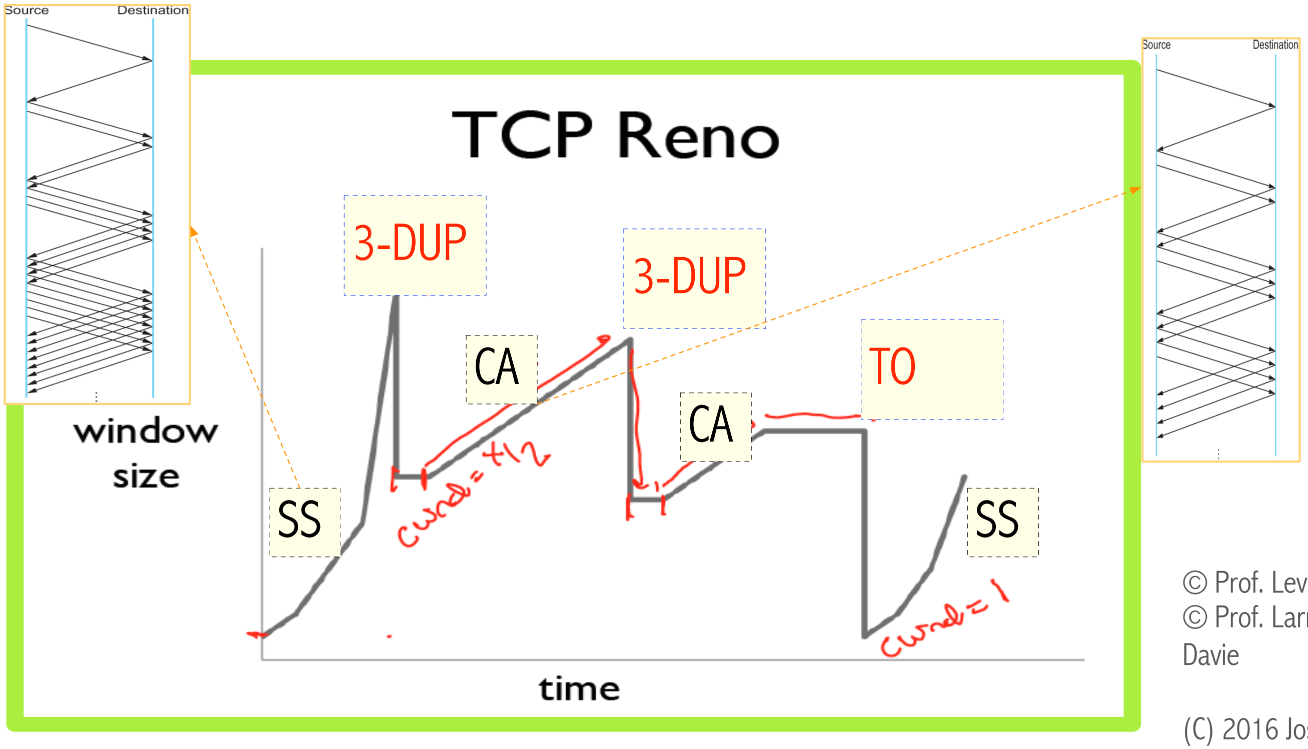




© Prof. Levis, Stanford University

How discovers network capacity

- Slow Start (SS)
  - Probe for network capacity by growing CW (Congestion window)
    CW = 2 * CW each Rtt
  - Initially, CW = 1
- 3-DUP causes transition to CA (Congestion Avoidance) with CW = SSthrsh / 2
- TO (Timeout) causes SS to start again
  - Linux implements TCP Reno and CUBIC congetion control



Reno, Fast Retransmit and Fast Recovery

- Fast Retransmit:
  - Upon a 3-DUP the transmitter will retransmit the missing segment, only
- Fast Recovery
  - Also, artificially increase CW = CW + 3 to compensate for the 3-DUP that didn't advance LastByteAcked and which, therefore, could not be used to spur the transmitter to transmit 3 new segments
  - Use the remaining, upcoming ACKS to keep the transmission pace
  - NO Slow Start in Reno upon 3-DUP