Computer Communication Networks

Midterm Review

ICEN/ICSI 416 – Fall 2016

Prof. Dola Saha

UNIVERSITY AT ALBANY
State University of New York

Instructions

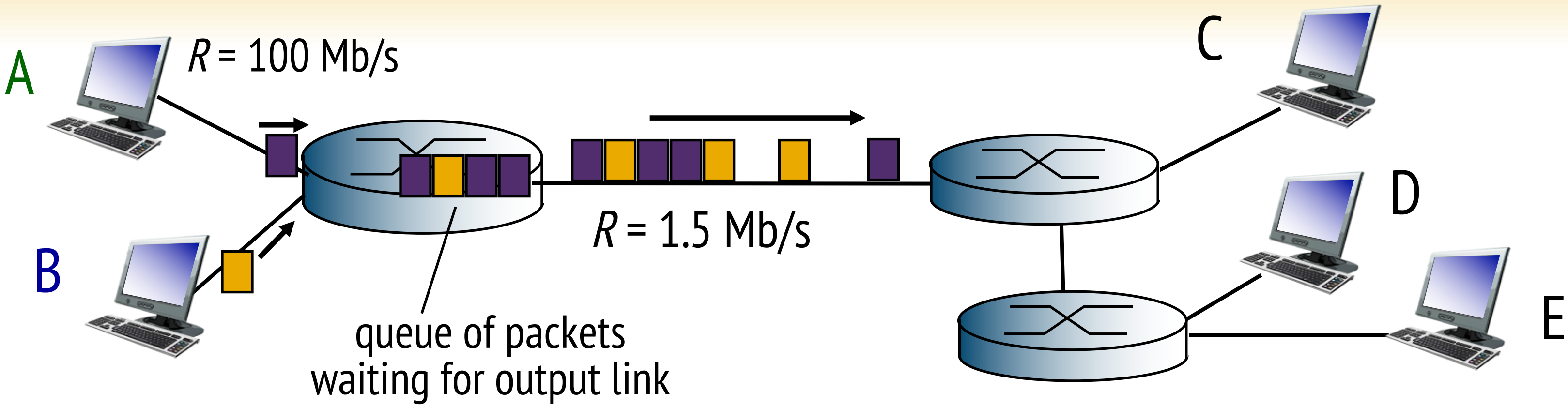
- Put your name and student id on each sheet of paper!
- The exam is closed book. You cannot use any computer or phone during the exam. You can use calculator, but not one from your phone or laptop.
- You have 60 minutes to complete the exam. Be a smart exam taker - if you get stuck on one problem go on to another problem.
- The total number of points for each question is given in parenthesis. There are 100 points total.
- Show all your work. Partial credit is possible for an answer, but only if you show the intermediate steps in obtaining the answer. If you make a mistake, it will also help the grader show you where you made a mistake.

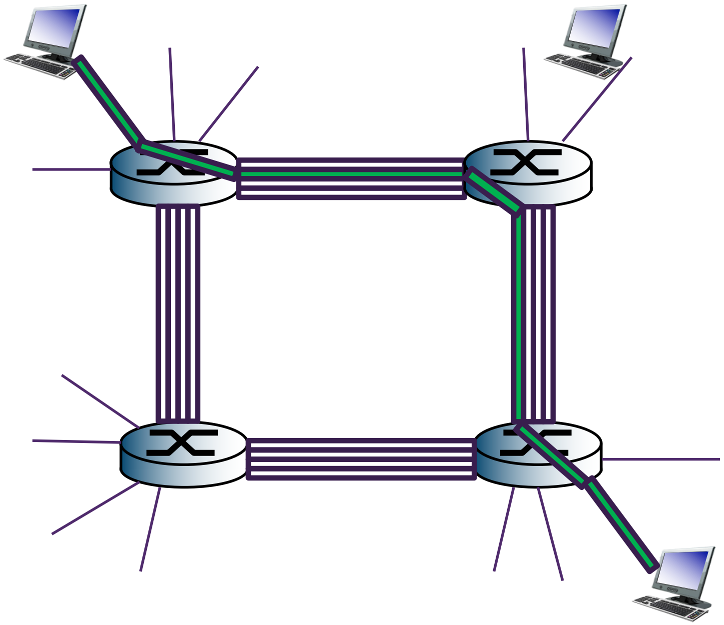
What is included?

- Foundation
- Application Layer
- Transport Layer

- The material covered by Prof. Hany Elgala will **NOT** be included in the midterm.

Packet Switching vs Circuit Switching

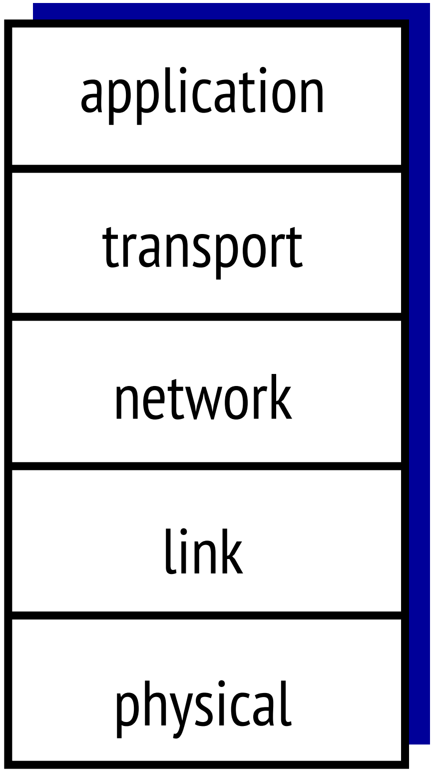




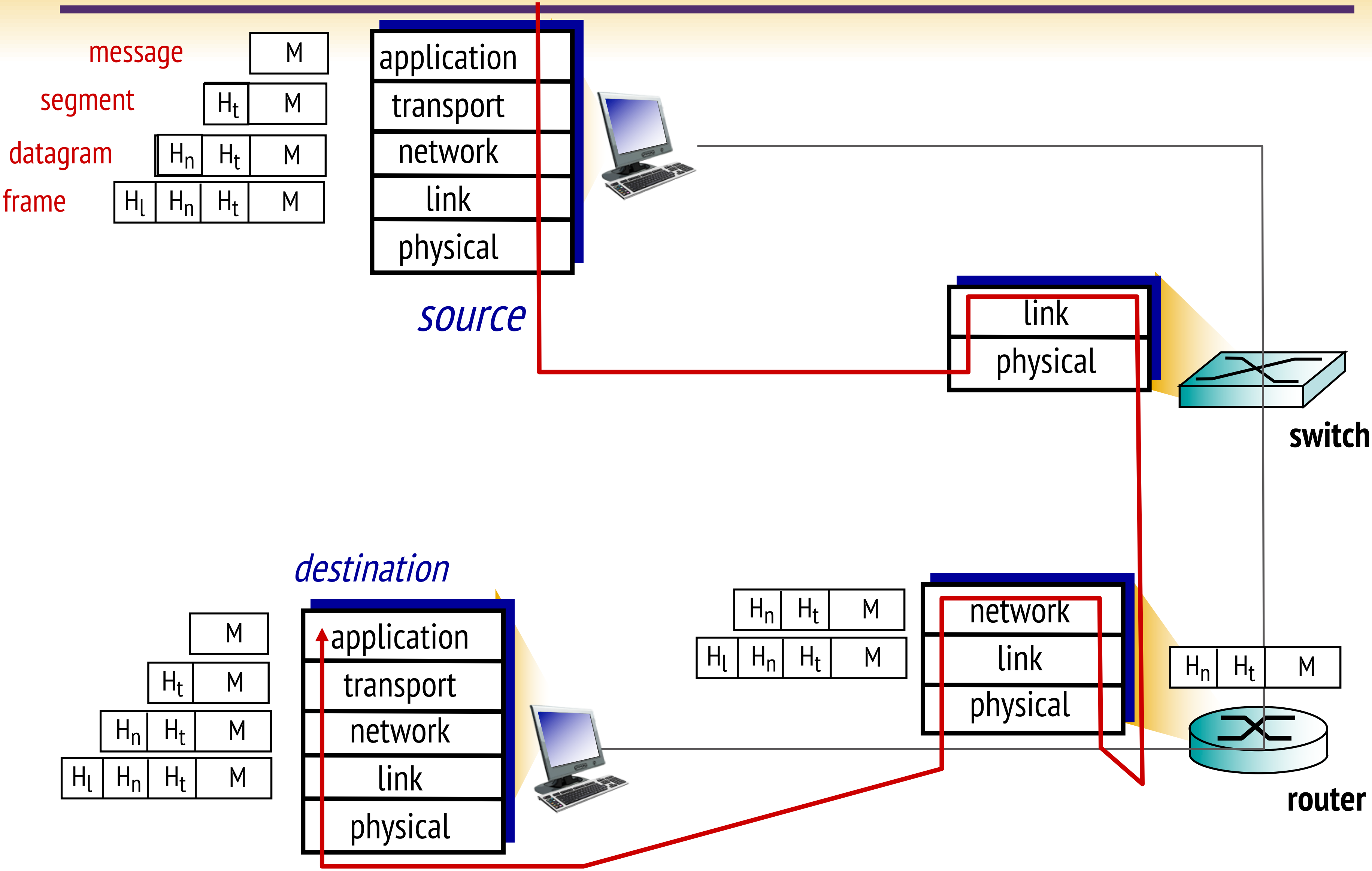
- Advantages
- Disadvantages

Internet Protocol Stack

- application: supporting network applications
  - FTP, SMTP, HTTP
- transport: process-process data transfer
  - TCP, UDP
- network: routing of datagrams from source to destination
  - IP, routing protocols
- link: data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi)
- physical: bits "on the wire" / "over the air"



Encapsulation



Socket

- What is a socket?
  - The point where a local application process attaches to the network
  - An interface between an application and the network
  - An application creates the socket

- The interface defines operations for
  - Creating a socket
  - Attaching a socket to the network
  - Sending and receiving messages through the socket
  - Closing the socket

Socket programming

Two socket types for two transport services:

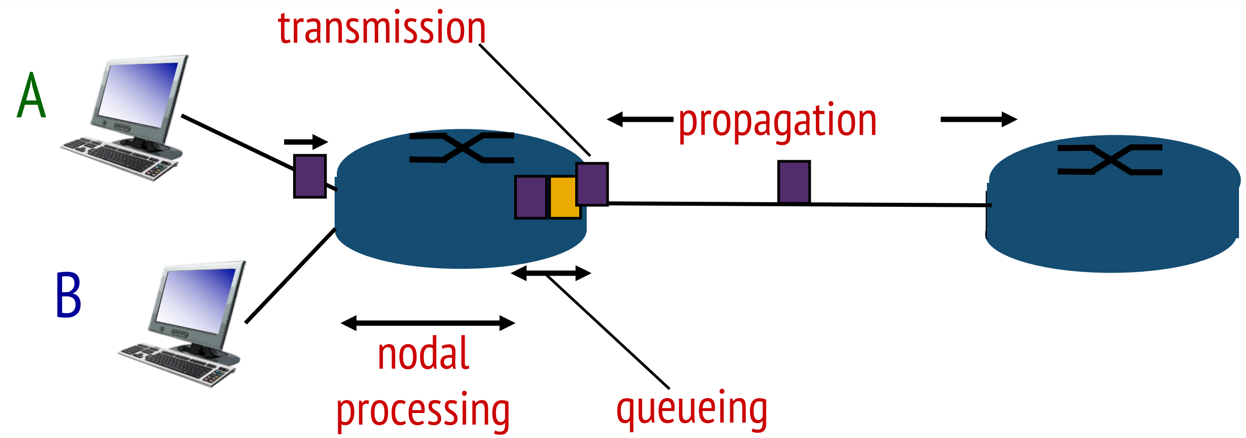
- *UDP:* unreliable datagram
- *TCP:* reliable, byte stream-oriented

- Server side:
  - DO NOT specify IP Address
  - By not specifying an IP Address, the application program is willing to accept connections on any of local hosts IP Addresses

Performance

- Bandwidth
  - Width of the frequency band
  - Number of bits per second that can be transmitted over a communication link
  - 1 Mbps: 1 x 10^6 bits/second = 1x2^{20} bits/sec
- Delay
  - Time elapsed for a packet to travel from a sender to receiver
  - seconds

Four Sources of Packet Delay



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

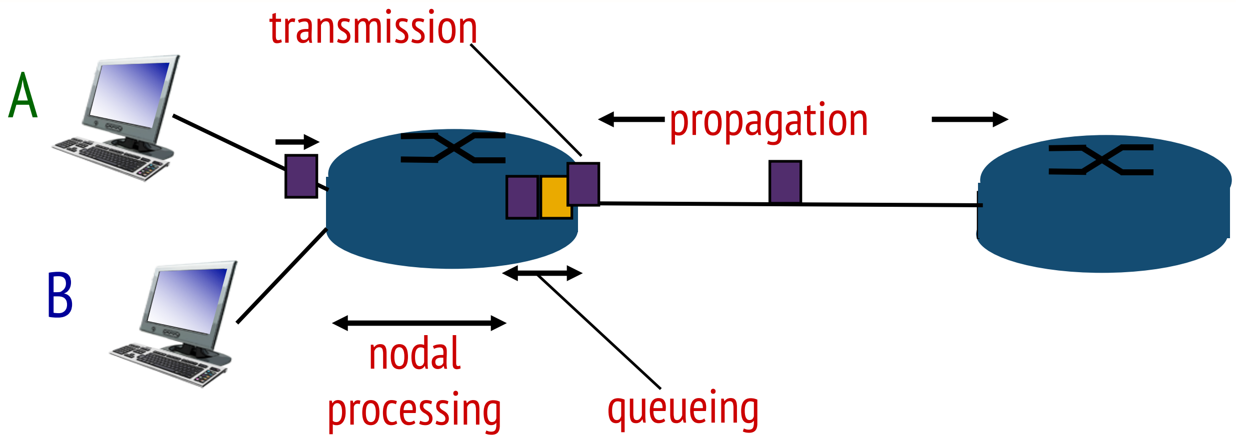
d_{proc}: nodal processing

- check bit errors
- determine output link
- typically < msec

d_{queue}: queueing delay

- time waiting at output link for transmission
- depends on congestion level of router

Four Sources of Packet Delay



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{trans}: transmission delay:

- L: packet length (bits)
- R: link *bandwidth (bps)*
- $d_{trans} = L/R$

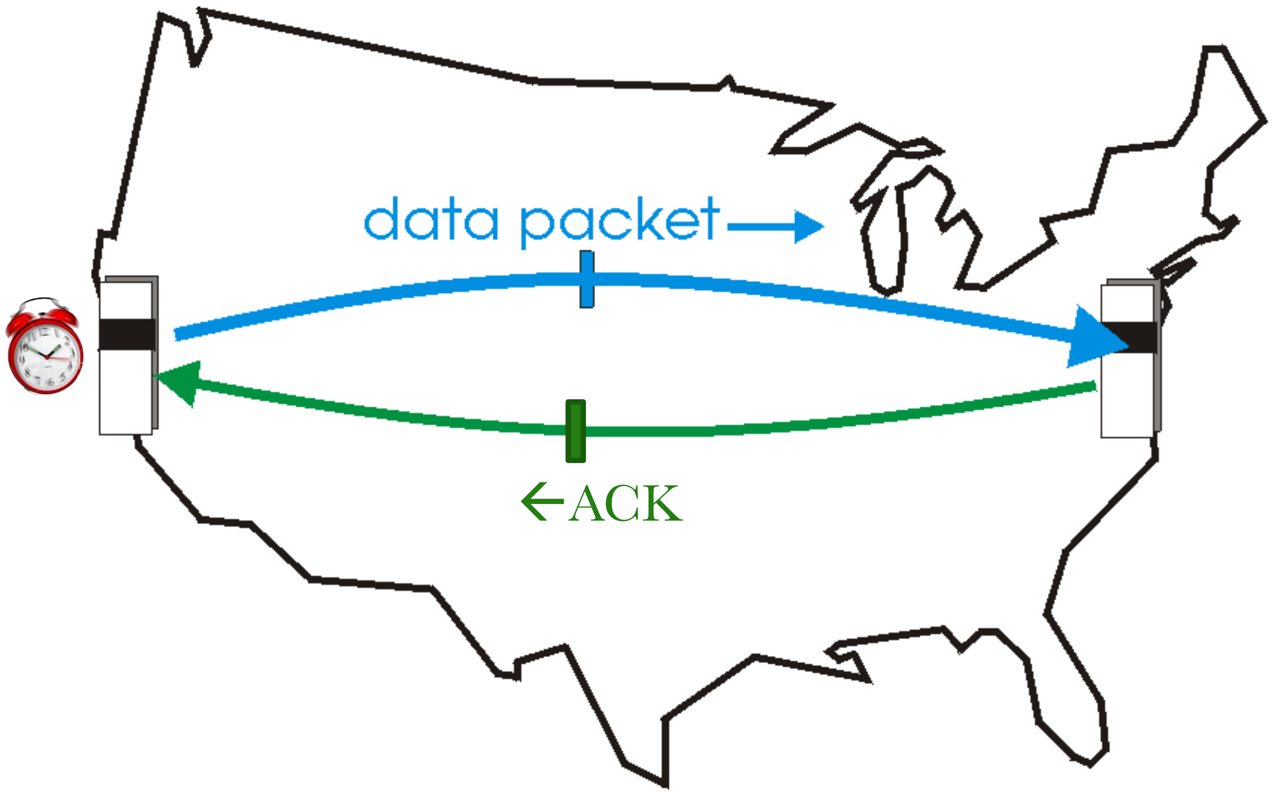
d_{trans} and d_{prop}
very different

d_{prop}: propagation delay:

- d: length of physical link
- s: propagation speed in medium (~2x10^8 m/sec)
- $d_{prop} = d/s$

Round Trip Time (RTT)

- Time:
  - From packet starting to leave a node
  - To response came back to the same node



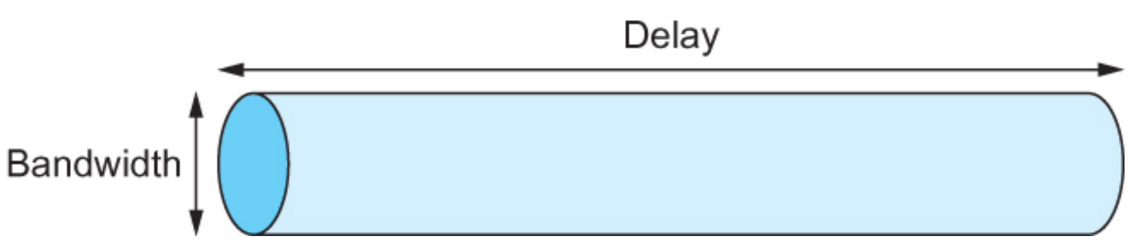
Performance

- Latency = Propagation + processing + transmit + queue
- Propagation = distance/speed of light
- Transmit = size/bandwidth
- Processing = depends on the node (hardware + software), but fairly constant
- Queue = congestion in the node → changes with time

- One bit transmission => propagation is important
- Large bytes transmission => bandwidth is important

Delay X Bandwidth

- We think the channel between a pair of processes as a hollow pipe
- Latency (delay) length of the pipe and bandwidth the width of the pipe
- Delay of 50 ms and bandwidth of 45 Mbps
  - 50×10^{-3} seconds x 45×10^{6} bits/second
  - 2.25×10^{6} bits = 280 KB data.
- Significance
  - This represents the maximum amount of data the sender can send before it would be possible to receive a response



Network as a pipe

Persistent and non-persistent HTTP

non-persistent HTTP issues:

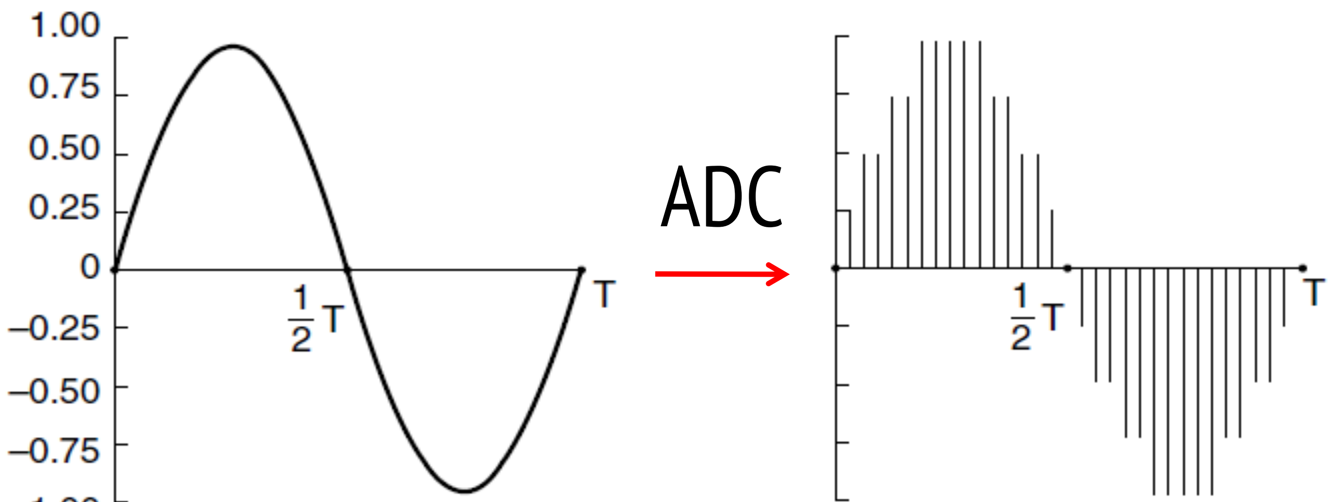
- requires 2 RTTs per object
- OS overhead for *each* TCP connection
- browsers often open parallel TCP connections to fetch referenced objects

persistent HTTP:

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

Digital Audio (1)

- ADC (Analog-to-Digital Converter) produces digital audio from a microphone
  - Telephone: 8000 8-bit samples/second (64 Kbps); computer audio is usually better quality (e.g., 16 bit)

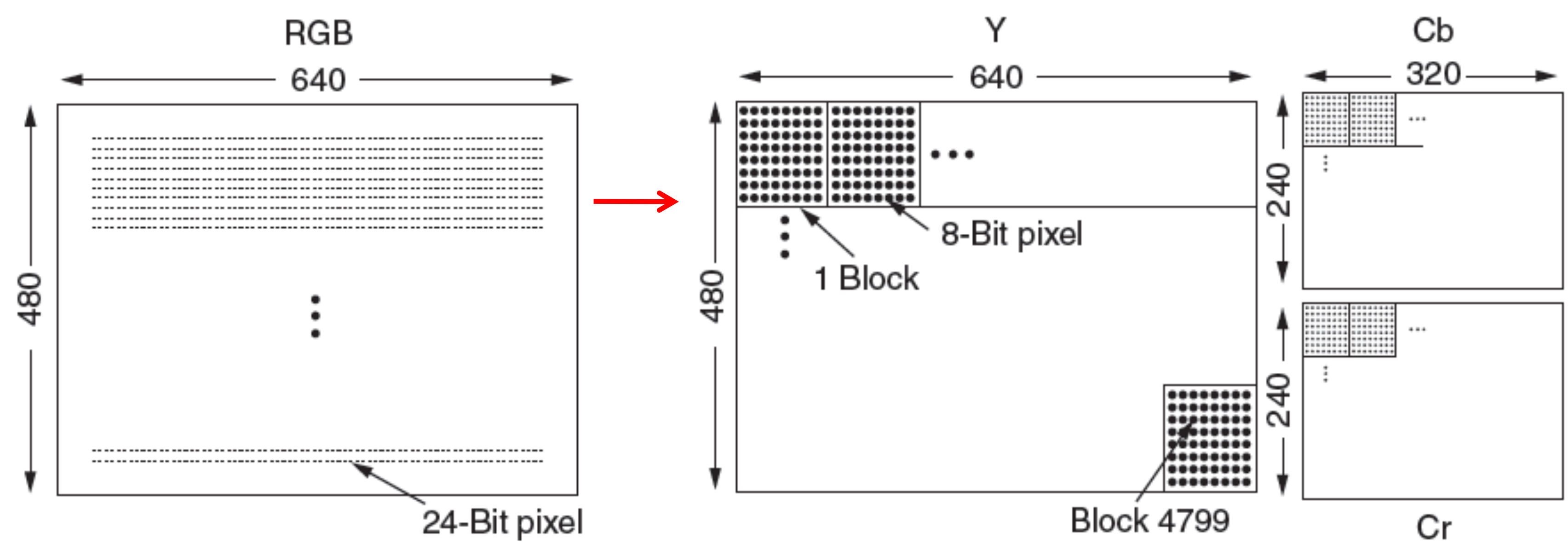


Continuous audio (sine wave)

Digital audio (sampled, 4-bit quantized)

Digital Video (3)

- Step 1: Pixels are mapped to luminance (brightness)/chrominance (YCbCr) color space
  - Luma signal (Y), Chroma signal: 2 components (Cb and Cr)
  - Chrominance is sub-sampled, the eye is less sensitive to chrominance



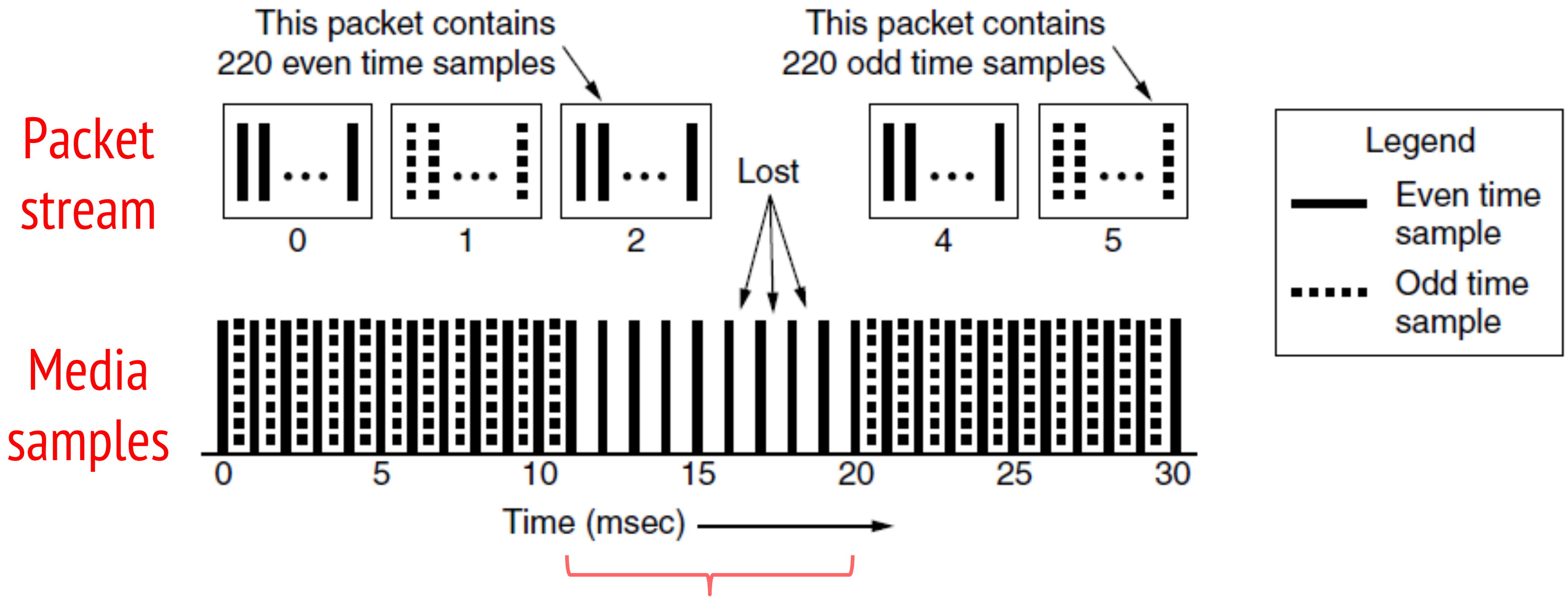
Input 24-bit RGB pixels

8-bit luminance pixels

8-bit chrominances for every 4 pixels

Streaming Stored Media (5)

- Interleaving spreads nearby media samples over different transmissions to reduce the impact of loss



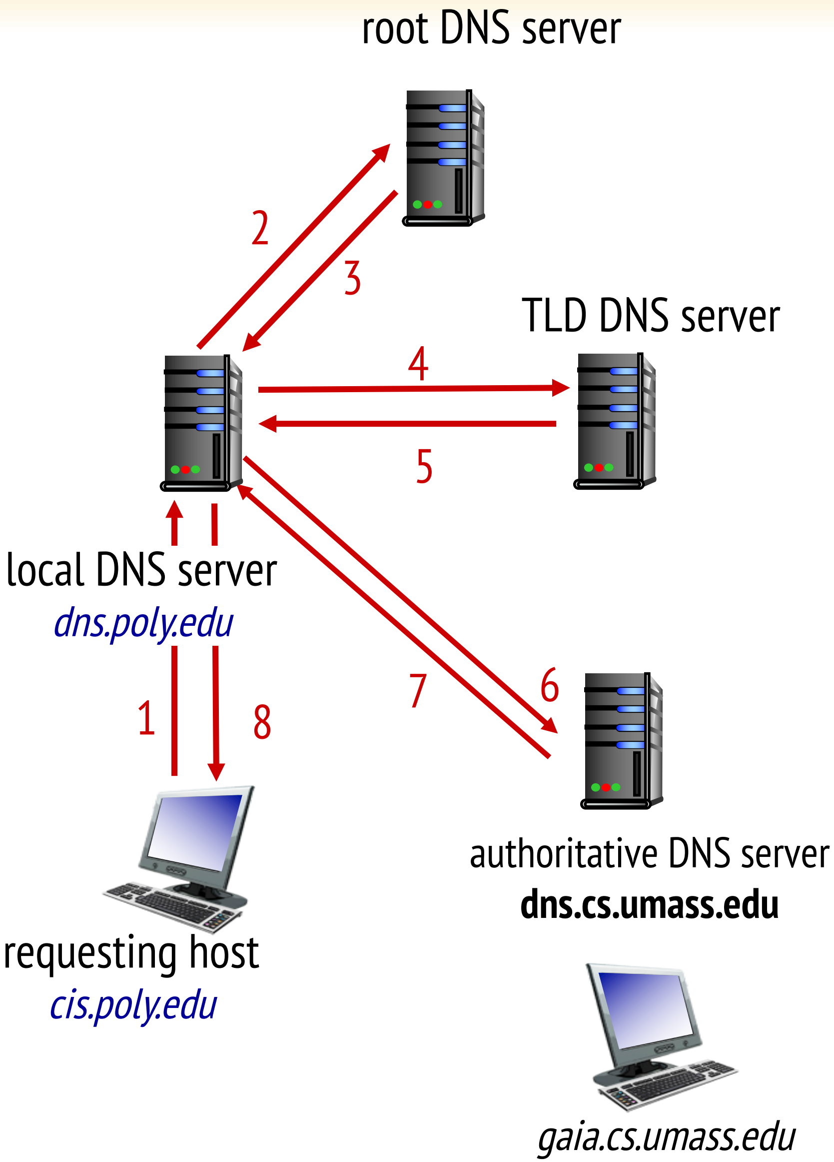
Loss reduces temporal resolution; doesn't leave a gap

DNS name resolution example

- host at cis.poly.edu wants IP address for gaia.cs.umass.edu

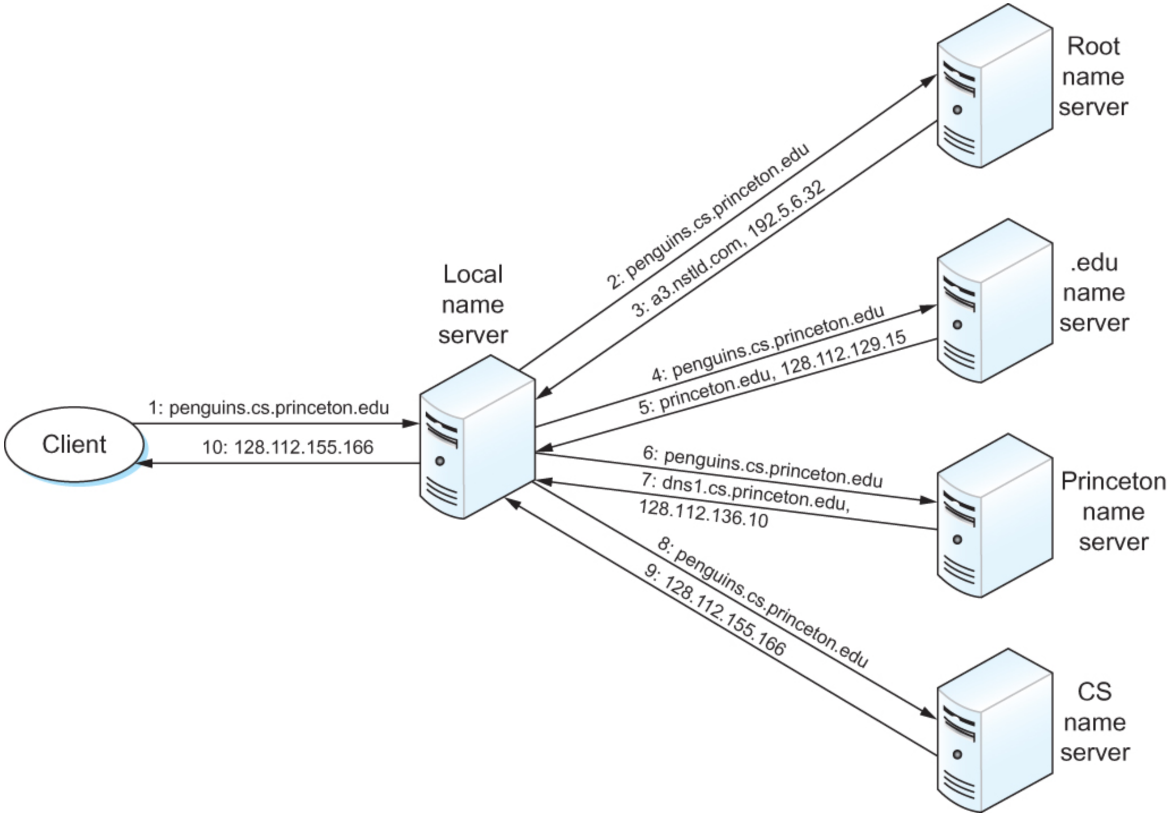
iterated query:

- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



Infrastructure Services

- Name Resolution



Name resolution in practice, where the numbers 1–10 show the sequence of steps in the process.

BitTorrent: requesting, sending file chunks

requesting chunks:

- at any given time, different peers have different subsets of file chunks
- periodically, Alice asks each peer for list of chunks that they have
- Alice requests missing chunks from peers, rarest first

sending chunks: tit-for-tat

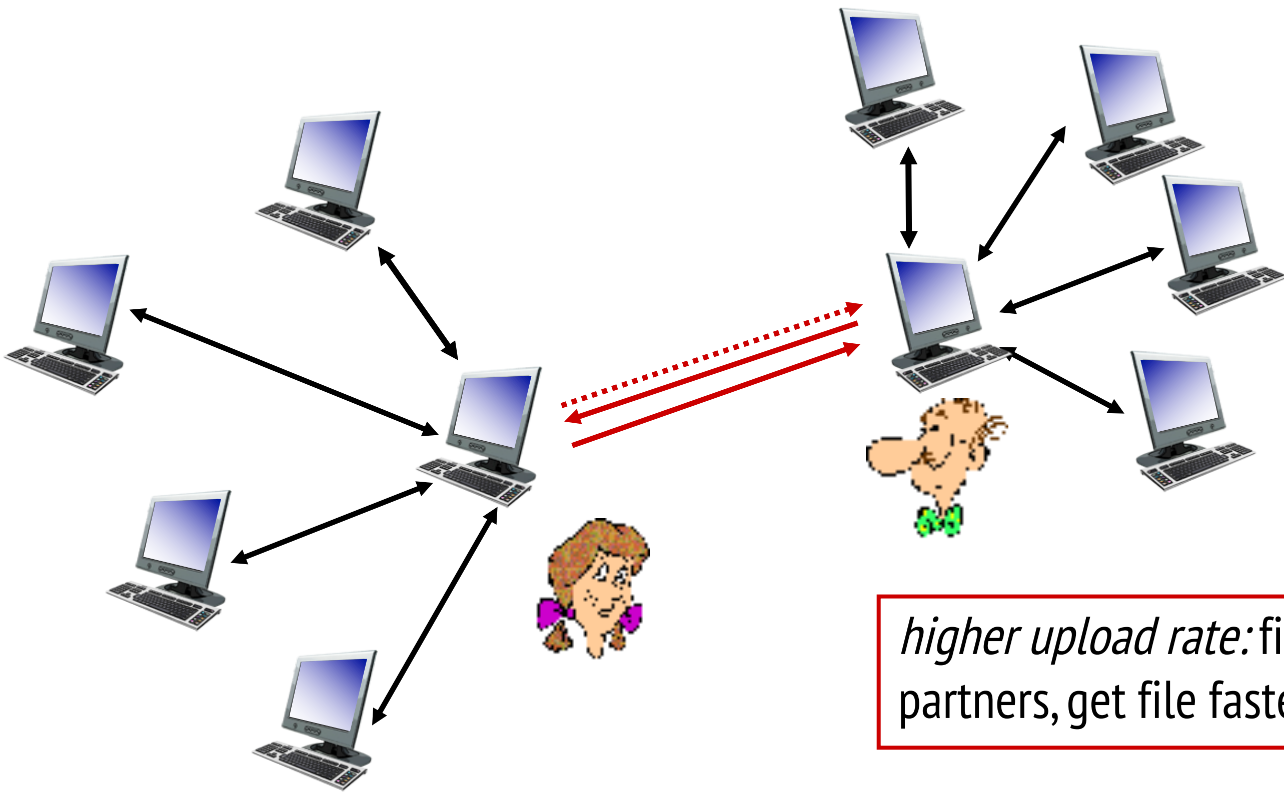
- Alice sends chunks to those four peers currently sending her chunks *at highest rate*
  - other peers are choked by Alice (do not receive chunks from her)
  - re-evaluate top 4 every 10 secs
- every 30 secs: randomly select another peer, starts sending chunks
  - "optimistically unchoke" this peer
  - newly chosen peer may join top 4

BitTorrent: tit-for-tat

(1) Alice "optimistically unchokes" Bob

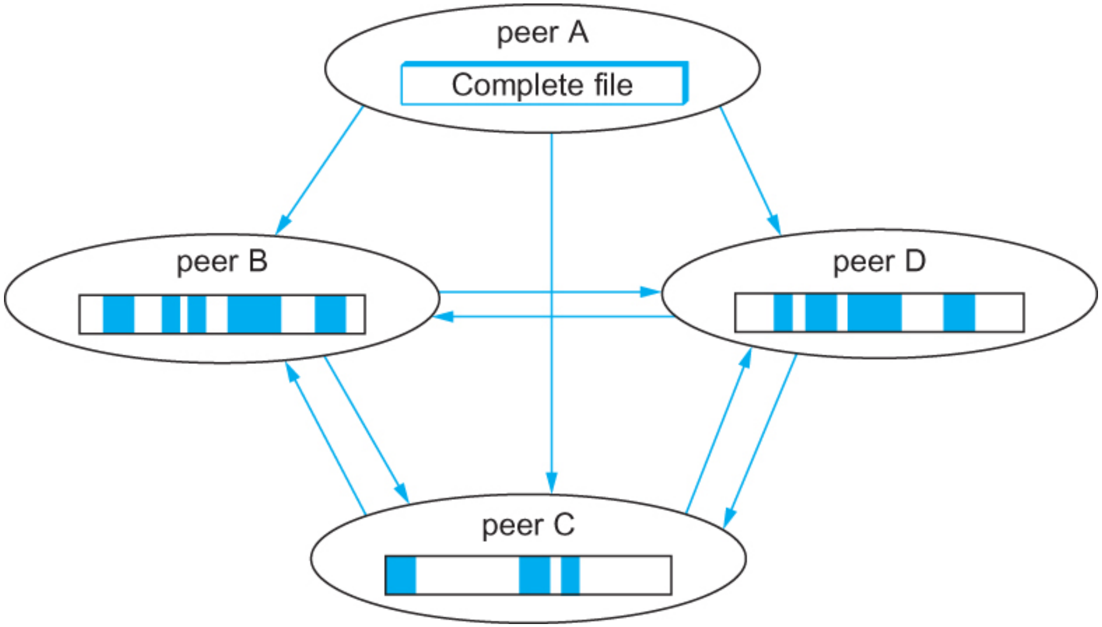
(2) Alice becomes one of Bob's top-four providers; Bob reciprocates

(3) Bob becomes one of Alice's top-four providers



higher upload rate: find better trading partners, get file faster !

BitTorrent: another aspect



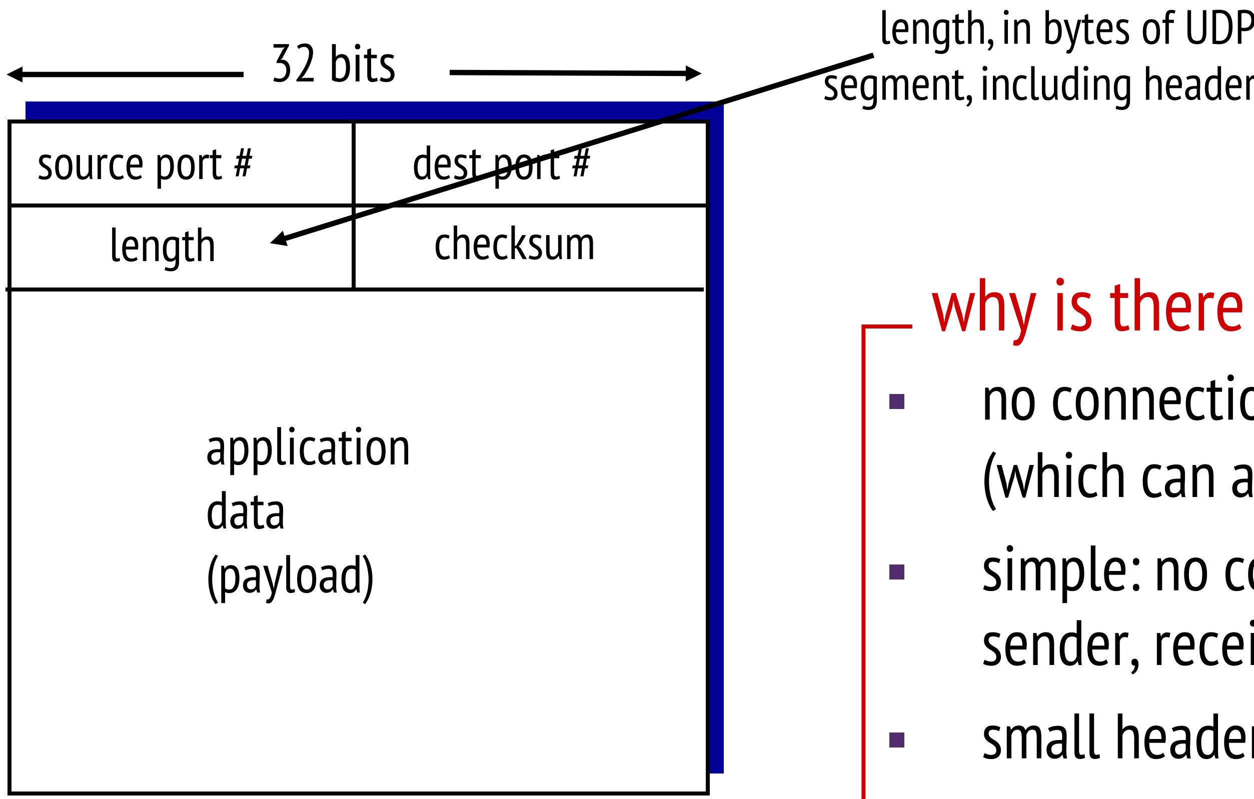
Peers in a BitTorrent swarm download from other peers that may not yet have the complete file

UDP: User Datagram Protocol [RFC 768]

- "no frills," "bare bones" Internet transport protocol
- "best effort" service, UDP segments may be:
  - lost
  - delivered out-of-order to app
- *connectionless:*
  - no handshaking between UDP sender, receiver
  - each UDP segment handled independently of others

- UDP use:
  - streaming multimedia apps (loss tolerant, rate sensitive)
  - DNS
  - SNMP
- reliable transfer over UDP:
  - add reliability at application layer
  - application-specific error recovery!

UDP: segment header

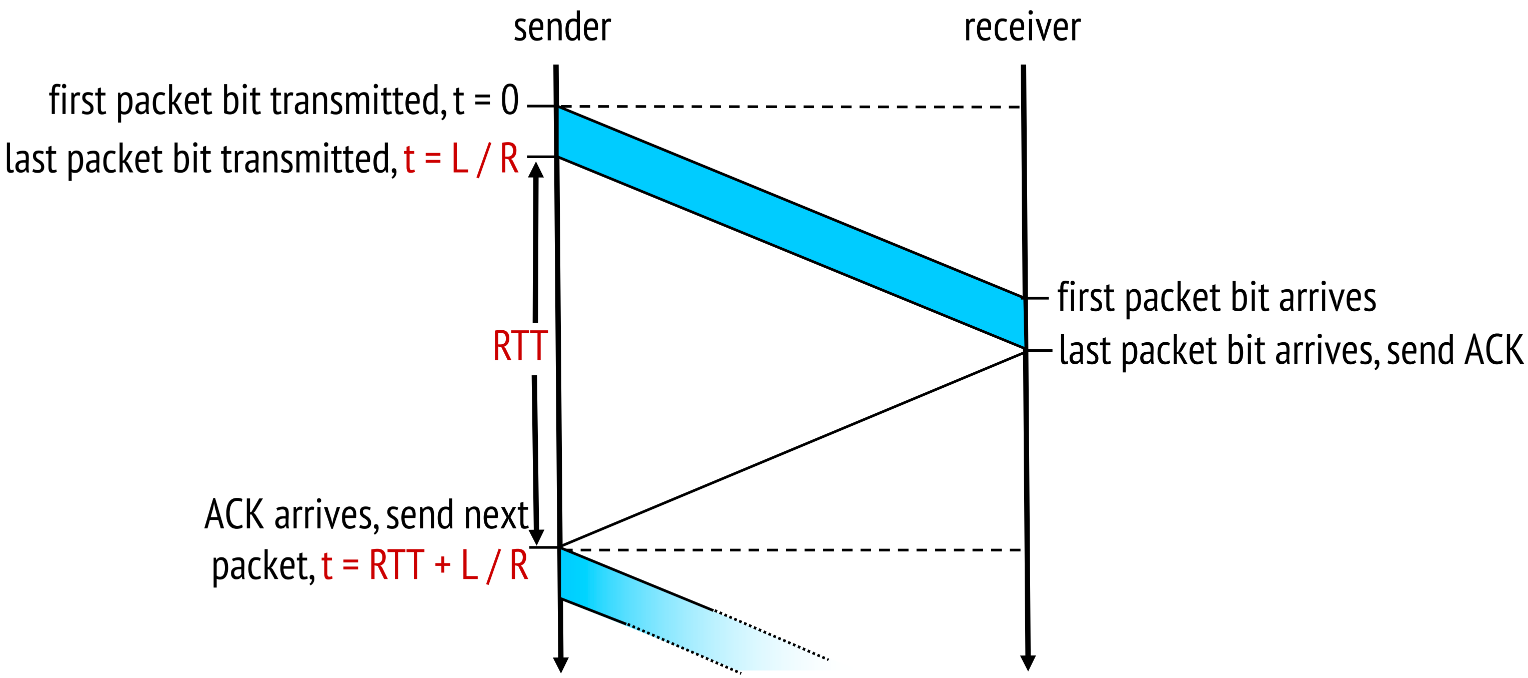


UDP segment format

why is there a UDP?

- no connection establishment (which can add delay)
- simple: no connection state at sender, receiver
- small header size
- no congestion control: UDP can blast away as fast as desired

rdt3.0: stop-and-wait operation

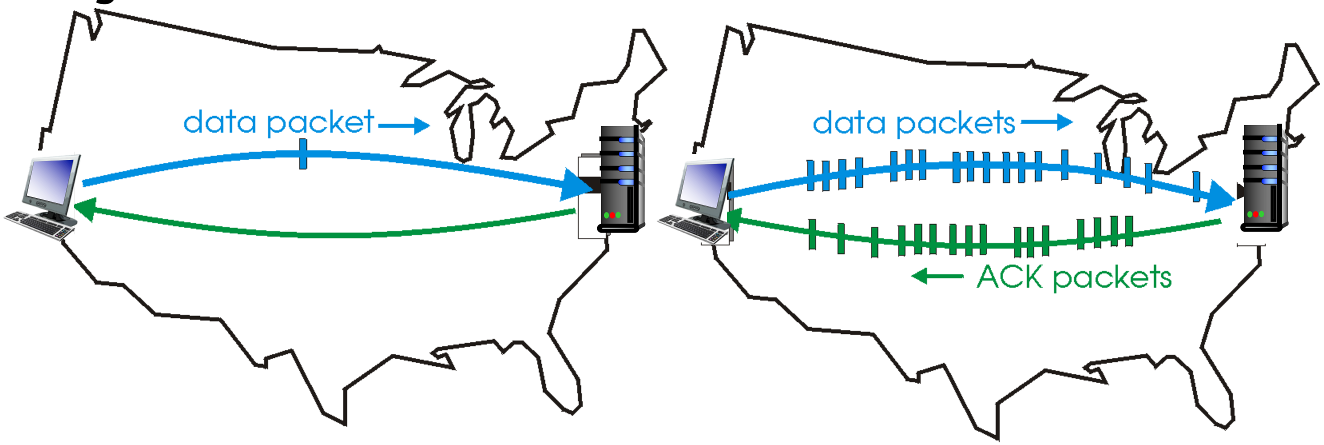


$$U_{sender} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

Pipelined protocols

pipelining: sender allows multiple, "in-flight", yet-to-be-acknowledged pkts

- range of sequence numbers must be increased
- buffering at sender and/or receiver

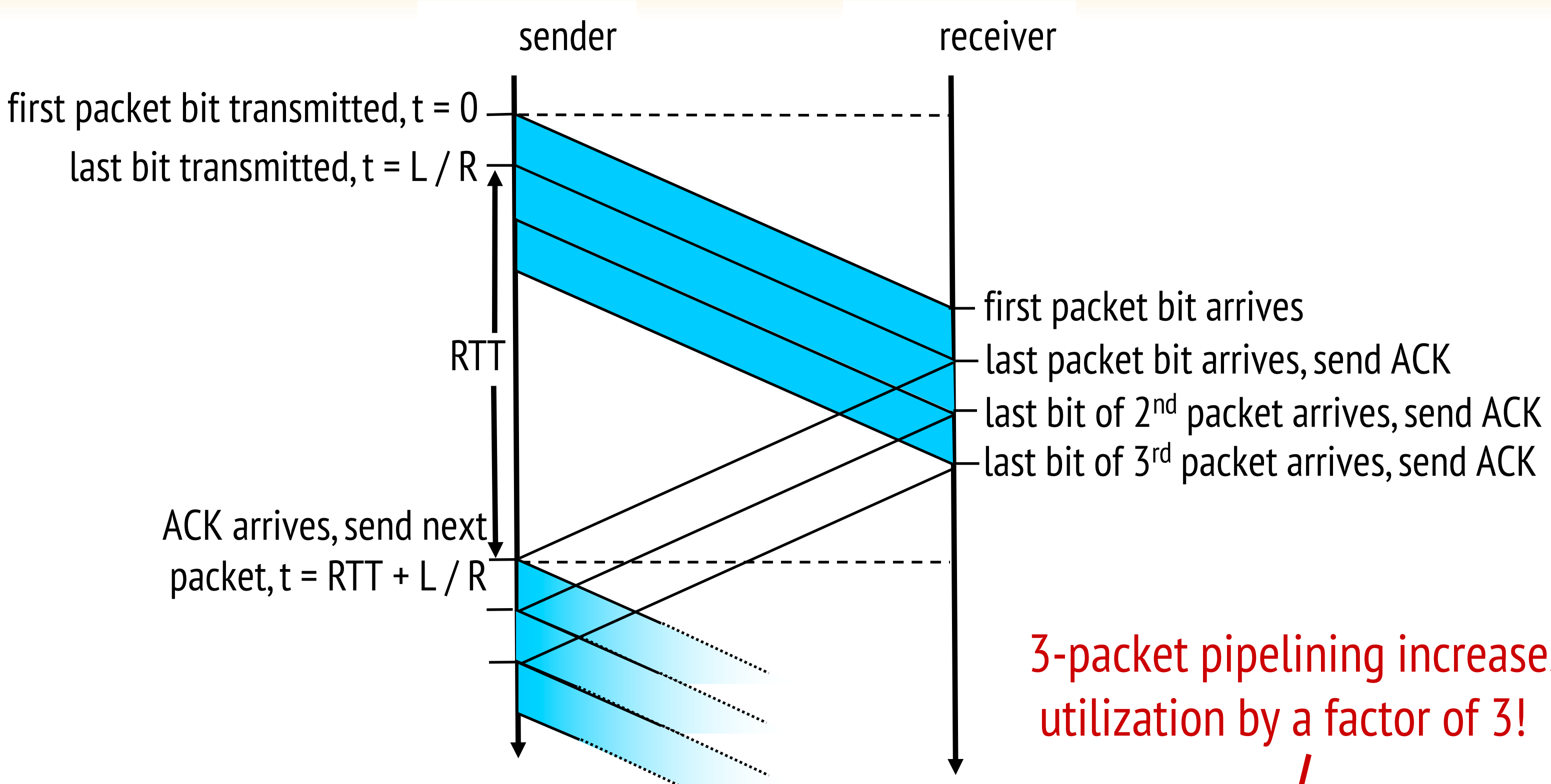


(a) a stop-and-wait protocol in operation

(b) a pipelined protocol in operation

- two generic forms of pipelined protocols: *go-Back-N, selective repeat*

Pipelining: increased utilization



3-packet pipelining increases utilization by a factor of 3!

$$U_{sender} = \frac{3L / R}{RTT + L / R} = \frac{.0024}{30.008} = 0.00081$$

Pipelined protocols: overview

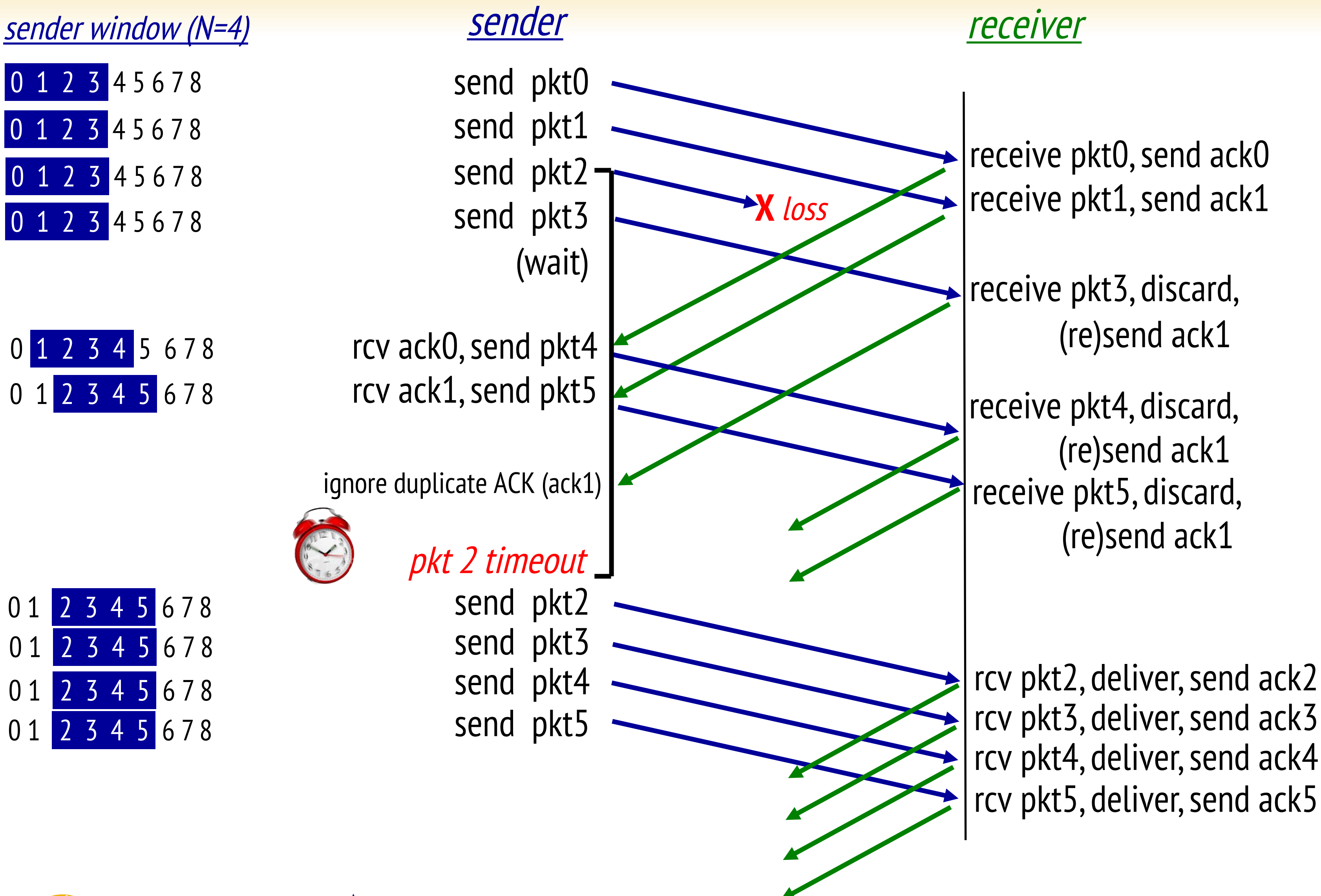
Go-back-N:

- sender can have up to N unacked packets in pipeline
- receiver only sends *cumulative ack*
  - doesn't ack packet if there's a gap
- sender has timer for oldest unacked packet
  - when timer expires, retransmit *all* unacked packets

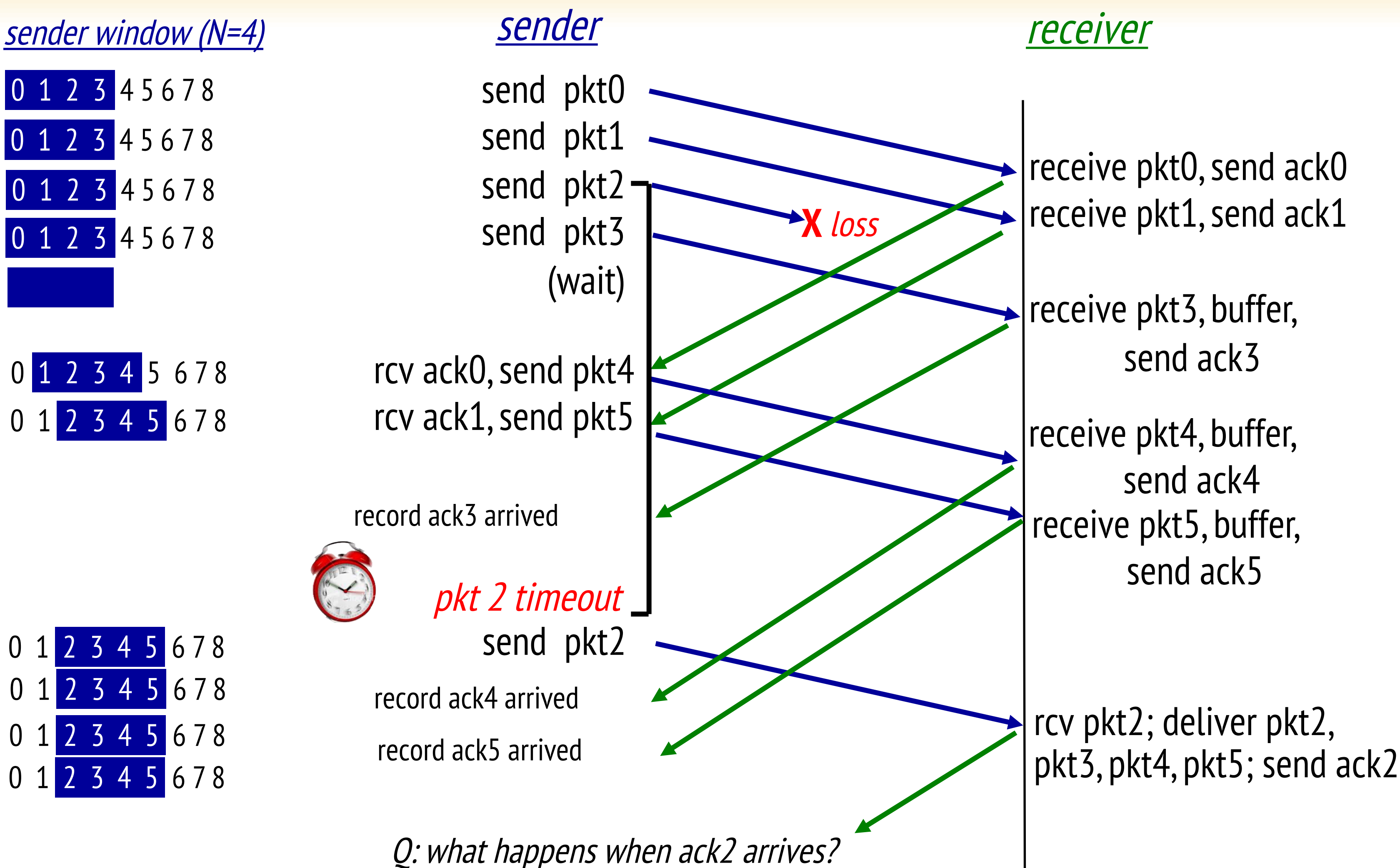
Selective Repeat:

- sender can have up to N unack'ed packets in pipeline
- rcvr sends *individual ack* for each packet
- sender maintains timer for each unacked packet
  - when timer expires, retransmit only that unacked packet

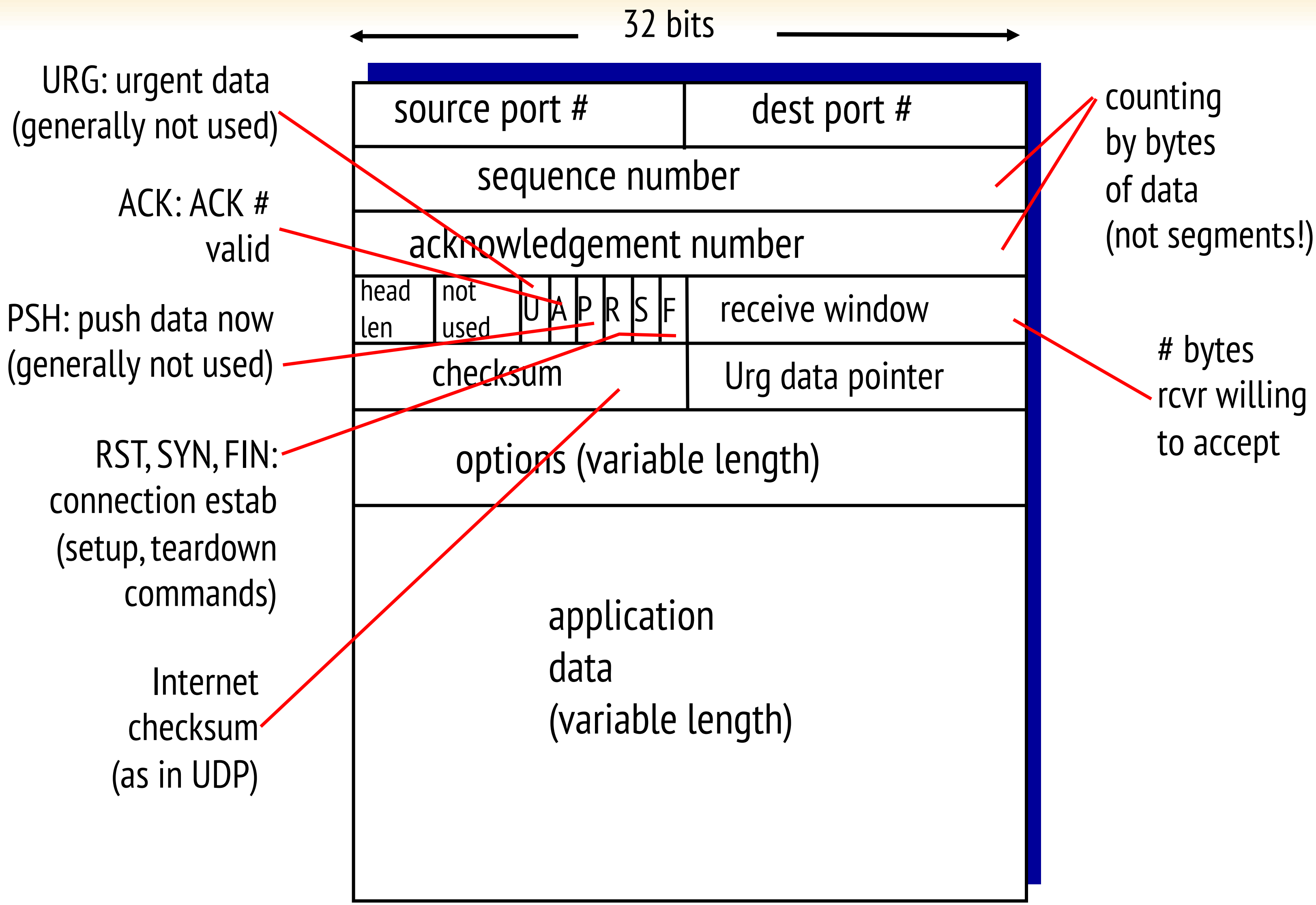
GBN in action



Selective repeat in action



TCP segment structure



TCP seq. numbers, ACKs

sequence numbers:

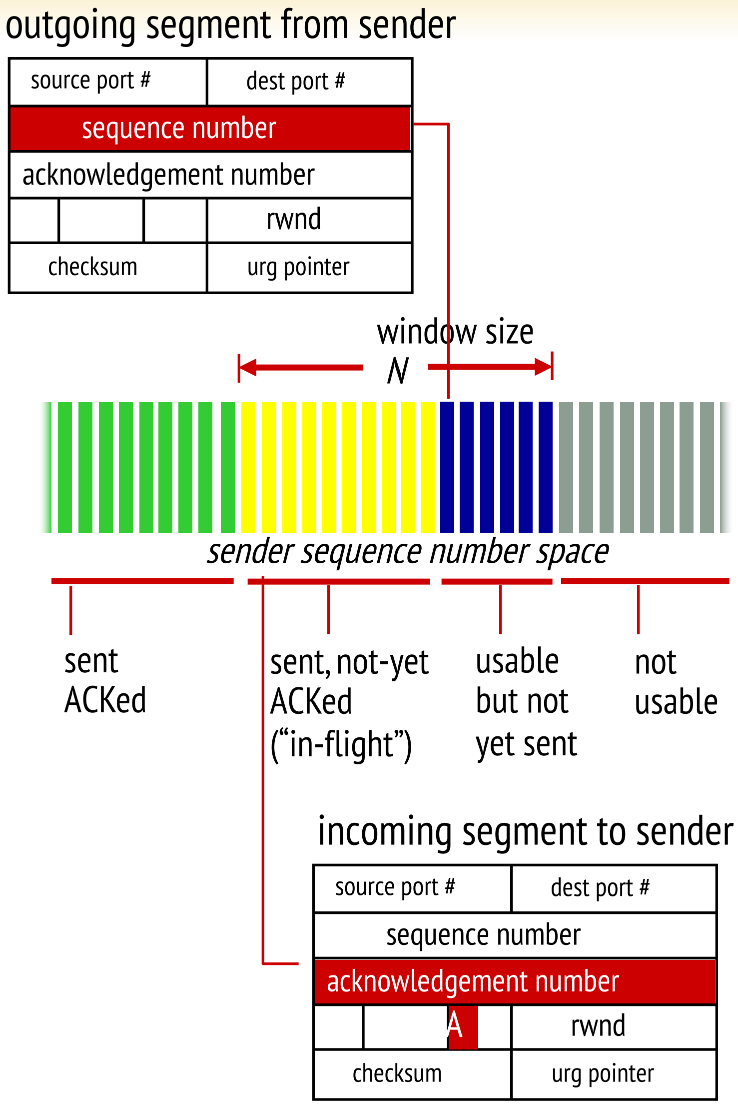
- byte stream "number" of first byte in segment's data

acknowledgements:

- seq # of next byte expected from other side
- cumulative ACK

Q: how receiver handles out-of-order segments

- A: TCP spec doesn't say, - up to implementor



TCP round trip time, timeout

Q: how to set TCP timeout value?

- longer than RTT
  - but RTT varies
- *too short:* premature timeout, unnecessary retransmissions
- *too long:* slow reaction to segment loss

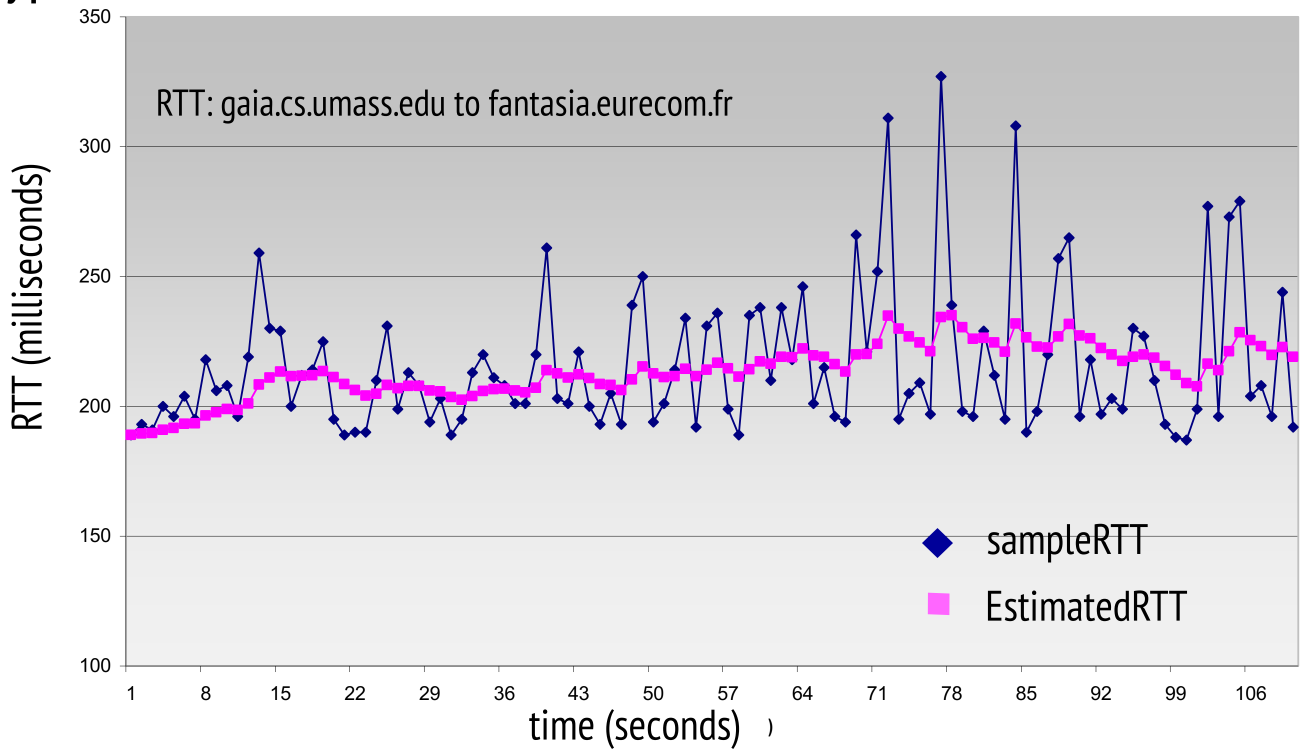
Q: how to estimate RTT?

- **SampleRTT**: measured time from segment transmission until ACK receipt
  - ignore retransmissions
- **SampleRTT** will vary, want estimated RTT “smoother”
  - average several *recent* measurements, not just current **SampleRTT**

TCP round trip time, timeout

EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT

- exponential weighted moving average
- influence of past sample decreases exponentially fast
- typical value: α = 0.125



Timeout = 2*EstimatedRTT

Jacobson/Karels Algorithm

- timeout interval: **EstimatedRTT** plus "safety margin"
  - large variation in **EstimatedRTT ->** larger safety margin
- estimate SampleRTT deviation from EstimatedRTT:
- RFC 6298

$$\text{DevRTT} = (1-\beta)*\text{DevRTT} + \beta*(|\text{SampleRTT}-\text{EstimatedRTT}|)$$

(typically, $\beta = 0.25$)

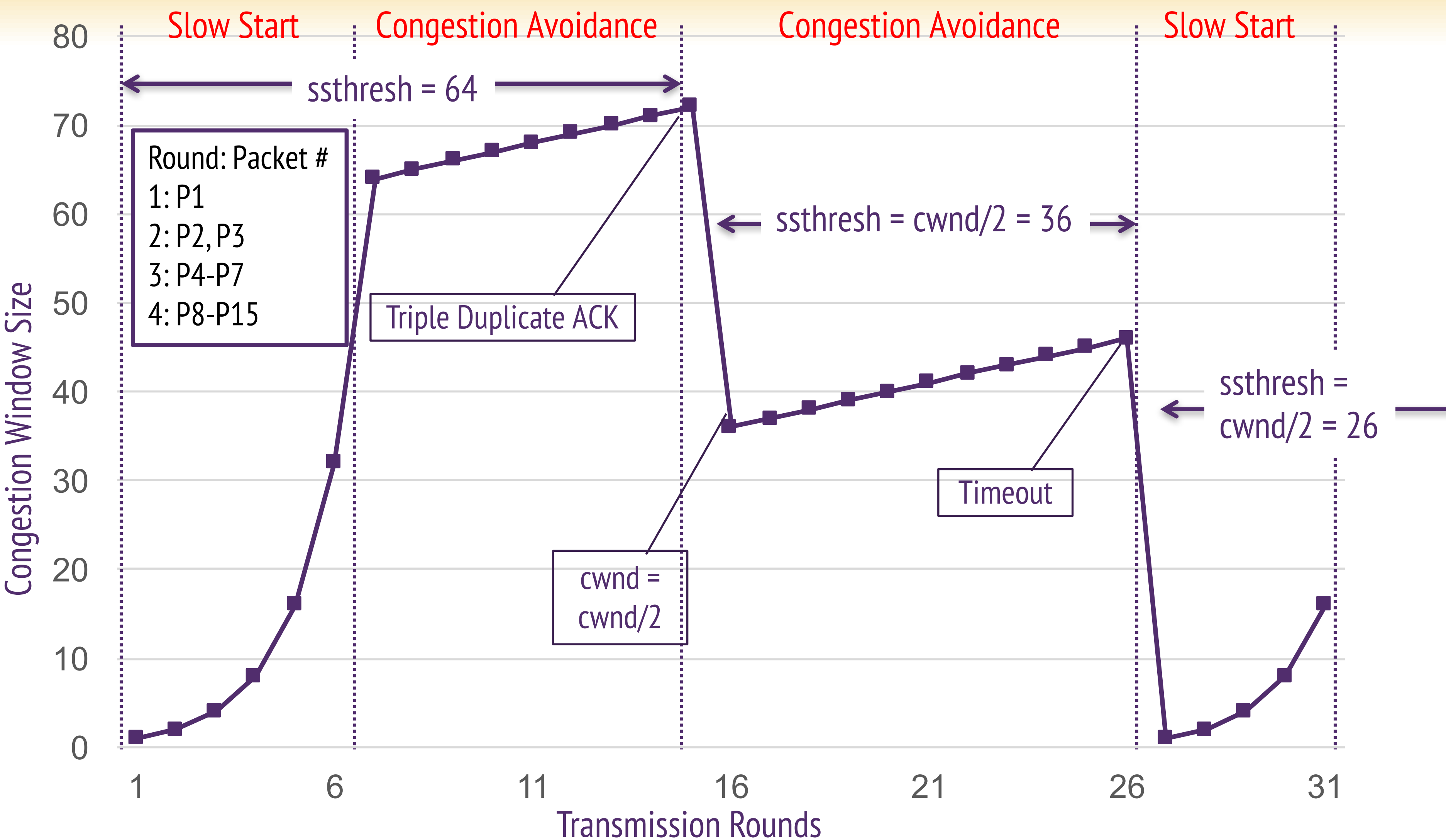
Measure of variability

$$\text{TimeoutInterval} = \text{EstimatedRTT} + 4*\text{DevRTT}$$

estimated RTT

"safety margin"

TCP Reno



Slow Start
Congestion Avoidance
Congestion Avoidance
Slow Start
ssthresh = 64
Round: Packet #
1: P1
2: P2, P3
3: P4-P7
4: P8-P15
Triple Duplicate ACK
ssthresh = cwnd/2 = 36
ssthresh = cwnd/2 = 26
Timeout
cwnd = cwnd/2
Congestion Window Size
80
70
60
50
40
30
20
10
0
1
6
11
16
21
26
31
Transmission Rounds

Good Luck!!!

UNIVERSITY AT ALBANY
State University of New York