hey!

LPE ERASEF
Definitions
Communication Protocols
Routing
→ Group Communication (multicast)
Mathematical Models
Automata
Hybrid Automata

Computation
Logic
Programming Languages
LPEs
Birman "Building Reliable and Secure Network Applications"
Manning 1996

Next week Chapter 13.10 - 13.12

Tanenbaum "Computer Networks"
Chapter 2.5

Tanenbaum "Distributed Operating Systems"
Prentice Hall 1995 Chapter 2
Physical layer: how to send 0s and 1s - electrical voltages
Connectors, ...

Data link: framing (packets)
error detection

Network: routing
Network: routing
Transport: Reliability
  Fragmentation/reassembly
  Fragmentation:
  break message into pieces
  * Packets may be dropped
  * Packets may be reordered
CRC - cyclic redundancy check
Checksum -
Transport -
Fragmentation

TCP (arbitrary size) transport
IP network (1K byte)

64K

1K

data  CRC

seq #

0  CRC

destination size
TCP - connection-oriented protocol
IP - connectionless

Sender send fragments with seq #
recv recv and reassemble

Sender: timeout
Receiver sends acknowledgments
ACK (Seq # i)

1. send seq i
2a. recv recvs seq i; sends Ack(i)

Sender: Sender recvs Ack(i)
If no Ack for a long time start over

2b. If ACK is lost
recv will get duplicate
go ahead and send ACK(i)

Proofs: Synchronous vs. asynchronous
bounded latency
Unbounded

Async: 
 sender recvs ACK(i)
eventually

Liveness: assumption
If a message is sent oo times it is eventually received oo times
Synch: bounds on transmission time

Liveness: if a message can be dropped
with $P(droop)$
or at most $N$ times

Sender: sliding window
Window size $N$

1. Sender sends up to $N$ fragments
2. Receiver acknowledges periodically
if $N_2$ messages received
with seq # in order
$ACK(10)$

If sender receives $NAK(i)$
it can move ahead, forget messages
$O - i \ N sent$

Sender buffer
$\text{Sender}$

Receiver
receives $msg \ i \ \ oops$
$\text{msg \ i+2}$

$NAK(i+1)$ negative acknowledgment
Sender gets $NAK(i)$, resends $msg \ i$
\[\begin{align*}
& A \land B \\
& A \lor B \\
& A \Rightarrow B \\
& \forall A, B \\
& \exists A, B
\end{align*}\]

- Point-to-point
- Point-to-multipoint
- Group comm / multicast protocols