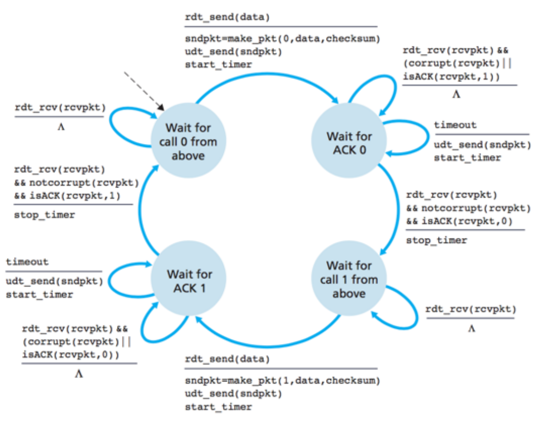
Chapter 3:



ure 3.15 ♦ rdt3.0 sender

Reliable Data Transfers:
Version 1: Sender send packet, Receiver receives it
Version 2.0: Add ACKS or NAKS so that sender knows that data was send properly
Version 2.1: Add sequence numbers to the ACK or NAK so that sender can send duplicate packets if it receives a corrupt ACK or NAK and the receiver can know whether the data it is receiving is new or a retransmission
Version 2.2: Get rid of the NAKs. If the sender receives two duplicate ACKS, it knows that the data packet it just sent was corrupted and sends it again.
Version 3.0: Add a timer, so that the receiver knows that a packet was possible lost.

Go Back N:
Sender:
If the window is not full, then it will send a packet with the next available window buffer. **If it receives an ACK**, it understands that it is a **cumulative acknowledgement** off all the packets including and before the one if received an ACK for, and moves the window up appropriately. **Timeout event:** Resends all the packets that have been previously sent but have not been asked
Receiver:
Sends the ACK of the last **successfully recieved, in order** packet. If it receives a packet out of order, then it discards it because it will be retransmitted anyway

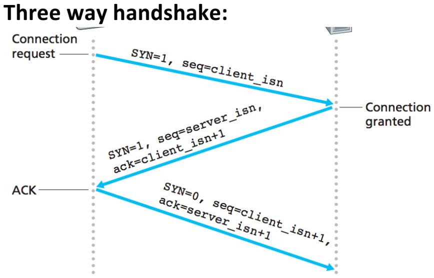
Selective Repeat:
Sender: If there is room in the window, the sender will send the packet with the next seq number**.** Each packet has its own individual timer. If a packets timer runs out, it is retransmitted**.** If it receives ACKs, then it marks those packets as ACKED and moves the window accordingly
Receiver:
If it receives a packet that is within its window, it sends an ACK back to the sender, moves the window accordingly and sends packets up to the application layer. If packet was already received, send an ACK anyway in case the previous ACK for it was lost. Otherwise, ignore the packets

TCP:
- The sequence number is the byte stream number of the first byte in the packet
- Cumulative Acknowledgement: TCP only acknowledges bytes up to the first missing bytes
- TCP has no rules about what to do with out of order segments
- Three Duplicate ACKs received for seq y, resend packet y

EstRTT = (1-a)EstRTT + a(SampRTT)
DevRTT = (1-B)DevRTT + B|SampRTT – EstRTT|
A = 0.125 B = 0.5
TimeoutInt = EstRTT + 4 * DevRTT

Flow Control: Rcvr lets the sender know its rwnd value, which is the size of the available space in the buffer. If the sender has as many unacked segments in its buffer are less than rwnd

Three way handshake:



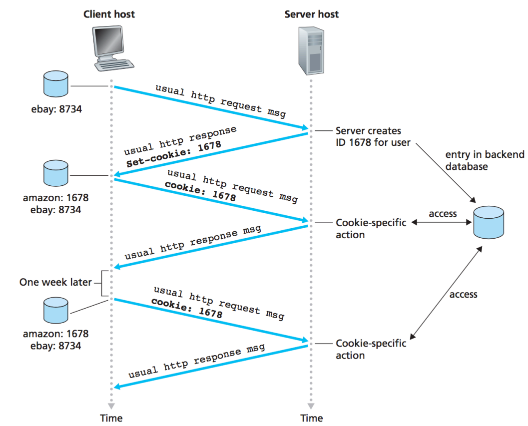
ATM ABR Congestion control:
A RM cell is sent with the data and then sent back by the rcvr.
EFCI bit: is set to 1 by a switch, letting the rcvr know that there is congestion
CI bit: Lets the rcvr know that there is a lot of congestion
NI bit: Let the recvr know that there is some delay
ER Setting: 2 bytes that allows the switch to set the minimum supportable rate

TCP Congestion Control:
Cwnd – the constraint on how much data can be sent in one RTT
Slow Start (exponential growth): add one MSS to the cwnd for each ACK recvd
Congestion control(linear increase): when you get to ssthresh, increase cwnd by one MSS every RTT.
TCP Reno:
If timeout occurs: set cwnd to 1 MSS again.
If triple ACK occurs:
Fast Retransmit: set the value of cwnd to half its value and increase cwnd incemenratly
(in TCP Tahoe, a triple ack would result in setting cwnd to 1 again)

$$\text{average throughput of a connection} = \frac{0.75 \cdot W}{RTT}$$

$$\text{average throughput of a connection} = \frac{1.22 \cdot MSS}{RTT\sqrt{L}}$$

CHAPTER 2:



Web Caching:

FTP: uses two parallel connections, a control connection and a data connection. Control is used to send commands while the data is used to actualy transfer the data between a server and a host
Commands = USER PASS LIST RETR STOR

Mail Access Protocols:
POP3: TCP connection bw user and mail server
Has three phases:
Authorization: username/password
Transaction: get mail, delete mail
Update: Quit (server deletes mail marked for deletion)

IMAP: create folders for messages and access only certain parts of the message at a time

Web based email: user communicated with a remote mail server using HTTP protocol

DNS:
Root: highest level, global scale
Top level: .org .edu .com
Authoratative: for each organizaiotn
Local DNS server: communicated with other types of DNS server

Type	Name	Value
A	Hostname	IP
NS	Domain	Auth DNS server
CNAME	Hostname	canonical
MX	Hostname	Canonical (mail)

P2P

$$D_{cs} = \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\}$$

$$D_{P2P} = \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\}$$

CHAPTER 1:

Packet Switches: L/R

Store and forward: do not transmit packet till it gets all bits

D(endtoend) – NL/R

Circuit Switching:

Frequency division muxing

Time Division Muxing:

Problem:

640,000 bits to transfer

using a TDM with 24 slots

bit rate of 1.536 Mbps

500 msec to get end to end connection

Each slot has transmission rate of:

1.536/24 = 64kbps

640,000 / 64 = 10secs

10 secs + 4 msec = 10.5secs

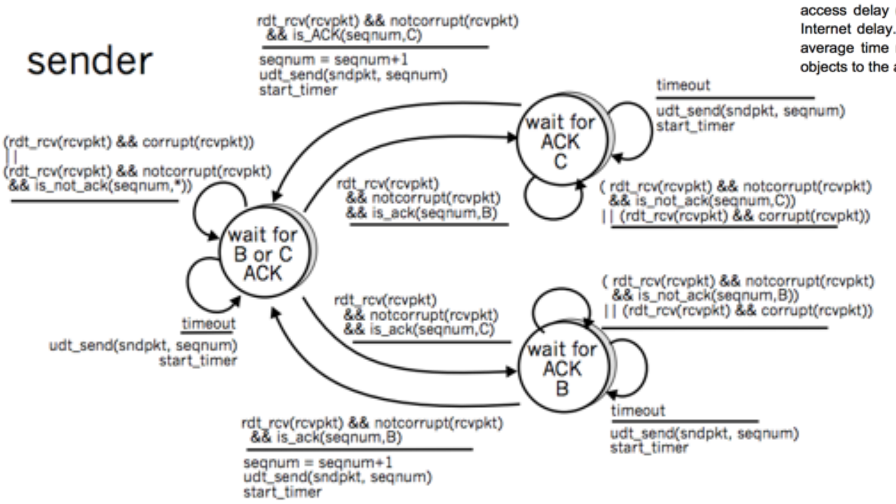
END

Delay(nodal) = d(processing) + d (queue) + d(transmission) + d(propagation)

Prop Delay: distance/time

Queue = La/R where a is arrival rate, L is bits/packet and R is bits/second

Nth backet has delay of (n-1)L/R secs



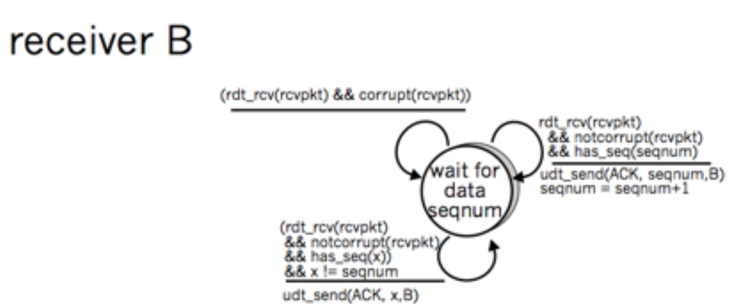


Figure 3. Sender and receiver for Problem 3.12

2. (20 points) Suppose users share a 2 Mbps link. Also suppose each user transmits continuously at 1 Mbps when transmitting, but each user transmits only 30 percent of the time. (See the discussion of statistical multiplexing in Section 1.3.)

a. When circuit switching is used, how many users can be supported?

b. For the remainder of this problem, suppose packet switching is used. Why will there be essentially no queuing delay before the link if two or fewer users transmit at the same time? Why will there be a queuing delay if three or more users transmit at the same time?

c. Find the probability that, at a given time, a given user is transmitting.

d. Suppose now there are totally four users. Find the fraction of time during which the queue grows.

Answer:

a: with circuit switching, each user needs to reserve a circuit with 1Mbps, therefore, only two users can be supported.

b: Since each user requires 1Mbps when transmitting, if two or fewer users transmit simultaneously, a maximum of 2Mbps will be required. Since the available bandwidth of the shared link is 2Mbps, there will be no queuing delay before the link. Whereas, if three users transmit simultaneously, the bandwidth required will be 3Mbps which is more than the available bandwidth of the shared link. In this case, there will be queuing delay before the link.

c. 0.3

d. The queue grows if and only if there are three or four users transmitting at the same time. Since the probability for a user transmitting is p=0.3, the probability that three

Problem 8 - (Chapter 2 problem 9, 10 points) Consider Figure 2.12 in your book, for which there is an institutional network connected to the Internet. Suppose that the average object size is 850,000 bits and that the average request rate from institution browsers to the origin servers is 16 requests per second. Also suppose that the amount of time it takes from when the router on the internet side of the access link forwards an HTTP request until it receives the response is three seconds on average. Model the total average response time as the sum of the average access delay (that is the delay from the Internet router to institution router) and the average Internet delay. For the average access delay use Delta/(1 - Delta * Beta). Where Delta is the average time required to send an object over the access link and Beta is the arrival rate of objects to the access link.

a. Find the total average response time.
b. Now suppose a cache installed in the institutional LAN. Suppose the miss rate is 0.4. Find the total response time.

a) The time to transmit an object of size L over a link or rate R is L/R. The average time is the average size of the object divided by R:

Δ = (850,000 bits)/(15,000,000 bits/sec) = .0567 sec

The traffic intensity on the link is given by βΔ=(16 requests/sec)(.0567 sec/request) = 0.907. Thus, the average access delay is (.0567 sec)/(1 - .907) = .6 seconds. The total average response time is therefore .6 sec + 3 sec = 3.6 sec.

b) The traffic intensity on the access link is reduced by 60% since the 60% of the requests are satisfied within the institutional network. Thus the average access delay is (.0567 sec)/[1 – (.4)(.907)] = .089 seconds. The response time is approximately zero if the request is satisfied by the cache (which happens with probability .6); the average response time is .089 sec + 3 sec = 3.089 sec for cache misses (which happens 40% of the time). So the average response time is (.6)(0 sec) + (.4)(3.089 sec) = 1.24 seconds. Thus the average response time is reduced from 3.6 sec to 1.24 sec