CIS-427 – Computer Networks and Disc Processes

With Professor Dr. Zheng Song

HW1

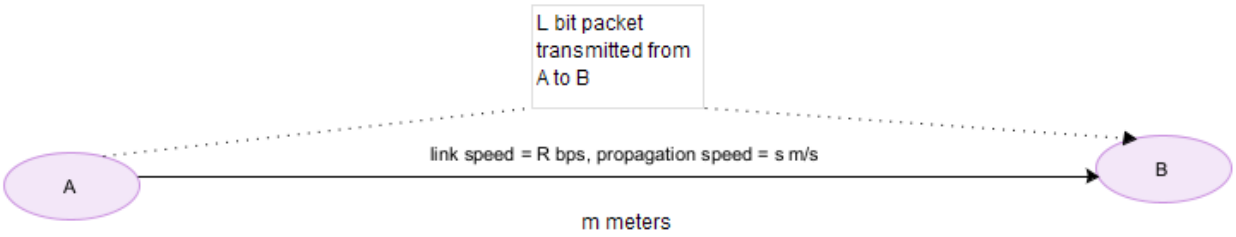
Demetrius Johnson

9-22-2022

Note we assume 1K = 1000, 1M = 1,000,000, 1G = 1,000,000,000, 1Byte = 8 bits

*note, I make all of my diagrams in Visual Paradigm.

Question 1



a) d_{prop} in terms of *m* and *s* → host separation *m* / propagation speed *s* → ***m/s*** → (meters/meters-per-second = seconds).

b) d_{trans} in terms of *L* and *R* → packet size of L bits / link transmission speed R bps → ***L/R*** →(bits / bps = seconds).

c) $Delay_{prop+trans} = d_{prop} + d_{trans} =$ ***m/s + L/R***

Question 2 –

- 5Mbps shared link
- 250kbps for each user during transmission
- Each user only transmits 10% of the time

a) With circuit switching, the bandwidth segment (using Frequency Division Multiplexing→FDM) for transmission quantity for each user must be exclusively reserved at all times (250kb dedicated, guaranteed resource for each user). Thus, take the bandwidth/transmission size to get the number of user system can support → 5Mbps / 250kbps = 5*10^6 / 250*10^3 = 20 users.

b) Probability that a given user is transmitting via packet switching: P(k users transmitting) = select k users from a pool of n users * probability k users out of n people are transmitting * probability n-k (remaining users in the pool n of users) are not transmitting = n!/k!(n-k)! * 0.1^k * (1-0.1)^(n-k). So, if you want to know the P(a given user(s) are transmitting), simply plug in values for n (number of user in the system) and k; in this case, for a *given* user, we plug in k=1 to get

   n!/1!(n-1)! * 0.1^1 * (1-0.1)^(n-1).

c) with 120 users (n=120), then

   P(k users transmitting simultaneously) = 120!/k!(120-k)! * 0.1^k * (1-0.1)^(120-k).

d) Following the logic from parts b and c, in order to find the P(21 or more users transmitting simultaneously), you need to SUM all of the probabilities of k users transmitting simultaneously, starting from k = 21, up to k=n →

   SUM_{k=21→k=n}[n!/k!(n-k)! * 0.1^k * (1-0.1)^(n-k)]

Alternatively, you could do 1 – SUM_{k=20→k=0}[n!/k!(n-k)! * 0.1^k * (1-0.1)^(n-k)]

Question 3 (8*10^6 bits = (8*10^6 / 10^6)Mbs = 8Mbs)

a) Without segmentation, it would take (considering only transmission speed of the link = 2Mbps) 8Mb/2Mbps = 4 seconds to reach the first switch. With each switch using store and forward packet switching, then it would take 3 hops to get from source to destination; thus 3 * 4 = 12 seconds total to get from source to destination.

b) With 800 10kb packets, it would take 10kb/2Mbps = 0.005s to move the first packet to the first switch. The second packet will be fully received at the first switch at time 0.005s*2 = 0.01s.

c) With message segmentation then, it would only take 800 packets * 0.005s of transmission time for each packet, with all switches able to transmit simultaneously upon receiving each 10kb packet + 2*0.005s for the time it takes for the last packet to be transmitted through the last 2 hops = 800*0.005s + 2*0.005s = 4.01s. This is about 3x faster than without message segmentation since all switches cannot do work simultaneously as they have to wait to receive one large message before they can transmit one at a time.

d) It is good to use message segmentation in order that when transmission fails or data gets corrupted, you do not have to retransmit the entire message, only parts of the message need to be retransmitted. This also greatly enhances network performance and speed.

e) Some of the drawbacks of message segmentation is that it can cause network congestion and thus packet loss and delay due to buffer overflows on the routers. This is because so much data can fully saturate the link to its full capacity and greater, which is good, but only if we control congestion via protocols.

Question 4 (the following tables generated using MS Excel programming; I was going to upload the spreadsheet but you do not allow anything but PDF files for upload)

- File size = 15Gbs
- N peers
- Server upload rate = 30Mbps
- Peer download rate *d_i* = 2Mbps, upload rate of u

Client-server Minimum Distribution Model (u is irrelevant, since only server is uploading)

i = N	File Size	Server Upload Rate	u_i	d_i	server transmission time (s) = N*F/server upload rate	client download time (s) = F/d	Distribution time (s) from server to clients = **max**{server trans time, client download time}
10	15Gb	30Mbps	any	2Mbps	5000	7500	7500
100	15Gb	30Mbps	any	2Mbps	50000	7500	50000
1000	15Gb	30Mbps	any	2Mbps	500000	7500	500000

Continue onto next page→

P2P Minimum Distribution Model (u = 300Kbps)

i = N	File Size	Server Upload Rate	u_i	d_i	server trans. time (s) for sending 1 file (the first transmission of the file into the P2P system) = F/server upload rate	client download time (s) = F/d	transmission time (s) for N files, distributed amongst server and all clients = NF/(server trans time + SUM[trans. time for all clients])	Distribution time (s) from server to clients = **max**{server trans time for 1 file (the first transmission into the system), client download time, time it takes to distribute N files using trans power of server and all clients combined}
10	15Gb	30Mbps	300Kbps	2Mbps	500	7500	4545	7500
100	15Gb	30Mbps	300Kbps	2Mbps	500	7500	25000	25000
1000	15Gb	30Mbps	300Kbps	2Mbps	500	7500	45455	45455

P2P Minimum Distribution Model (u = 700Kbps)

i = N	File Size	Server Upload Rate	u_i	d_i	server trans. time (s) for sending 1 file (the first transmission of the file into the P2P system) = F/server upload rate	client download time (s) = F/d	transmission time (s) for N files, distributed amongst server and all clients = NF/(server trans time + SUM[trans. time for all clients])	Distribution time (s) from server to clients = **max**{server trans time for 1 file (the first transmission into the system), client download time, time it takes to distribute N files using trans power of server and all clients combined}
10	15Gb	30Mbps	700Kbps	2Mbps	500	7500	4054	7500
100	15Gb	30Mbps	700Kbps	2Mbps	500	7500	15000	15000
1000	15Gb	30Mbps	700Kbps	2Mbps	500	7500	20548	20548

P2P Minimum Distribution Model (u = 2Mbps)

i = N	File Size	Server Upload Rate	u_i	d_i	server trans. time (s) for sending 1 file (the first transmission of the file into the P2P system) = F/server upload rate	client download time (s) = F/d	transmission time (s) for N files, distributed amongst server and all clients = NF/(server trans time + SUM[trans. time for all clients])	Distribution time (s) from server to clients = **max**{server trans time for 1 file (the first transmission into the system), client download time, time it takes to distribute N files using trans power of server and all clients combined}
10	15Gb	30Mbps	2Mbps	2Mbps	500	7500	3000	7500
100	15Gb	30Mbps	2Mbps	2Mbps	500	7500	6522	7500
1000	15Gb	30Mbps	2Mbps	2Mbps	500	7500	7389	7500

Essentially, I notice that for the Client-server model, u (transmission = upload rate) does not affect the distribution time of a file in the system. Also, server upload (transmission) time is the dominating factor for distribution time in the client-server model as the number of peers in the system (N) grows. As compared to the P2P distribution model, P2P demonstrates that as N gets larger, total distribution time increases too, but not as dramatically as with client-server model, since users help the server to upload (transmit) N Files to the system. As upload (transmission) speeds increase among N peers, then the time for distribution drops dramatically, up to the point where the bottleneck becomes the transmission time of the first file transmission, which is done by the server.