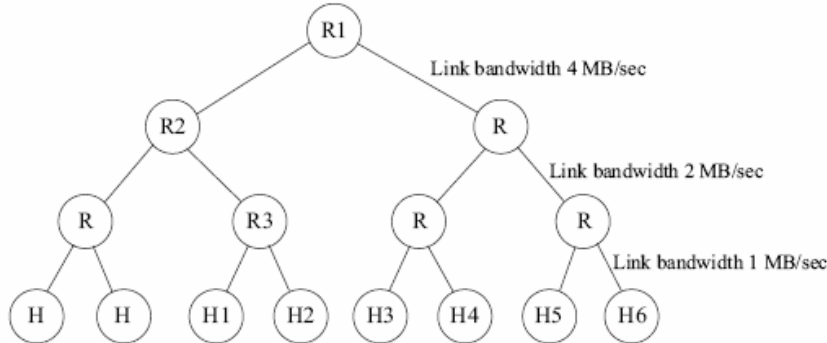


Solutions to Homework#5

6. R1 cannot become congested because traffic arriving at one side is all sent out the other, and the bandwidths on each side are the same.

We now show how to congest only the router R2 that is R1's immediate left child; other R's are similar.



We arrange for H3 and H4 to send 1MB/sec to H1, and H5 and H6 to send 1MB/sec to H2. Each of the links to the right of R1 reaches its maximum capacity, as does the R1—R2 link,

but none of these routers becomes congested. However, R2 now wants to send 4MB/sec to R3, which it cannot.

R3 is not congested as it receives at 2MB/sec from R2 and this traffic is evenly divided between H1 and H2.

10. (a) First we calculate the finishing times F_i . We don't need to worry about clock speed here since we may take $A_i = 0$ for all the packets. F_i thus becomes just the cumulative per-flow size, ie $F_i = F_{i-1} + P_i$.

Packet	size	flow	F_i
1	100	1	100
2	100	1	200
3	100	1	300
4	100	1	400
5	190	2	190
6	200	2	390
7	110	3	110
8	50	3	170

We now send in increasing order of F_i :

Packet 1, Packet 7, Packet 8, Packet 5, Packet 2, Packet 3, Packet 6, Packet 4.

- (b) To give flow 2 a weight of 2 we divide each of its F_i by 2, ie $F_i = F_{i-1} + P_i/2$; again we are using the fact that there is no waiting.

Packet	size	flow	weighted F_i
1	100	1	100
2	100	1	200
3	100	1	300
4	100	1	400
5	190	2	95
6	200	2	195
7	110	3	110
8	50	3	170

Transmitting in increasing order of the weighted F_i we send as follows:
 Packet 5, Packet 1, Packet 7, Packet 8, Packet 6, Packet 2, Packet 3, Packet 4.

16. (a) In slow start, the size of the window doubles every RTT. At the end of the i th RTT, the window size is 2^i KB. It will take 10 RTTs before the send window has reached 2^{10} KB = 1 MB.
- (b) After 10 RTTs, 1023 KB = 1 MB – 1 KB has been transferred, and the window size is now 1 MB. Since we have not yet reached the maximum capacity of the network, slow start continues to double the window each RTT, so it takes 4 more RTTs to transfer the remaining 9MB (the amounts transferred during each of these last 4 RTTs are 1 MB, 2 MB, 4 MB, 1 MB; these are all well below the maximum capacity of the link in one RTT of 12.5 MB). Therefore, the file is transferred in 14 RTTs.
- (c) It takes 1.4 seconds (14 RTTs) to send the file. The effective throughput is (10MB / 1.4s) = 7.1MBps = 57.1Mbps. This is only 5.7% of the available link bandwidth.
47. (a) If we start with an empty bucket but allow the bucket volume to become negative (while still providing packets), we get the following table of bucket “indebtedness”: At T=0, for example, we withdraw 5 packets and deposit 2.

Time, secs	0	1	2	3	4	5
Bucket volume	-3	-6	-5	-3	-7	-6

We thus need an initial bucket depth of 7, so as not to run out at T=4. Because all the volumes above are negative, the bucket with depth 7 never overflows.

- (b) If we do the same thing as above we get

Time, secs	0	1	2	3	4	5
Bucket volume	-1	-2	1	5	3	6

A bucket depth of 2 will thus accommodate T=1. If we start with an initially full bucket of depth 2, we get

Time, secs	0	1	2	3	4	5
Bucket volume	1	0	2	2	0	2

Entries in *italic* represent filling the bucket, and it turns out that due to this truncation we scrape the bottom at T=4 as well. However, the depth of 2 does suffice.