CH3

20.

a) True.

b) True. By essentially the same scenario as in (c).

c) True. Suppose the sender has a window size of 3 and sends packets 1, 2, 3 at $t0$. At $t1$ $(t1 > t0)$ the receiver ACKS 1, 2, 3. At $t2$ $(t2 > t1)$ the sender times out and resends 1, 2, 3. At $t3$ the receiver receives the duplicates and re-acknowledges 1, 2, 3. At $t4$ the sender receives the ACKs that the receiver sent at $t1$ and advances its window to 4, 5, 6. At $t5$ the sender receives the ACKs 1, 2, 3 the receiver sent at $t2$. These ACKs are outside its window.

d) True. Note that with a window size of 1, SR, GBN, and the alternating bit protocol are functionally equivalent. The window size of 1 precludes the possibility of out-of-order packets (within the window). A cumulative ACK is just an ordinary ACK in this situation, since it can only refer to the single packet within the window.

22.

In order to avoid the scenario of Figure 3.27, we want to avoid having the leading edge of the receiver's window (i.e., the one with the "highest" sequence number) wrap around in the sequence number space and overlap with the trailing edge (the one with the "lowest" sequence number in the sender's window). That is, the sequence number space must be large enough to fit the entire receiver window and the entire sender window without this overlap condition. So - we need to determine how large a range of sequence numbers can be covered at any given time by the receiver and sender windows.

Suppose that the lowest-sequence number that the receiver is waiting for is packet m. In this case, it's window is [m,m+w-1] and it has received (and ACKed) packet m-1 and the w-1 packets before that, where w is the size of the window. If none of those w ACKs have been yet received by the sender, then ACK messages with values of [m-w,m-1] may still be propagating back. If no ACKs with these ACK numbers have been received by the sender, then the sender's window would be [m-w,m-1].

Thus, the lower edge of the sender's window is m-w, and the leading edge of the receivers window is m+w-1. In order for the leading edge of the receiver's window to not overlap with the trailing edge of the sender's window, the sequence number space must thus be big enough to accommodate 2w sequence numbers. That is, the sequence number space must be at least twice as large as the window size, $k \geq 2w$.

31.

If the arrival rate increases beyond R/2 in Figure 3.46(b), then the total arrival rate to the queue exceeds the queue's capacity, resulting in increasing loss as the arrival rate increases. When the arrival rate equals R/2, 1 out of every three packets that leaves the queue is a retransmission. With increased loss, even a larger fraction of the packets leaving the queue will be retransmissions. Given that the maximum departure rate from the queue for one of the sessions is R/2, and given that a third or more will be transmissions as the arrival rate increases, the throughput of successfully deliver data can not increase beyond λ_{out}. Following similar reasoning, if half of the packets leaving the queue are retransmissions, and the maximum rate of output packets per session is R/2, then the maximum value of λ_{out} is (R/2)/2 or R/4.

32.

a) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 16, the congestion windows size is 42. Hence the threshold is 21 during the 18th transmission round.

b) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 22, the congestion windows size is 26. Hence the threshold is 13 during the 24th transmission round.

c) TCP slowstart is operating in the intervals [1,6] and [23,26]

d) The congestion window and threshold will be set to half the current value of the congestion window (8) when the loss occurred. Thus the new values of the threshold and window will be 4.

e) After the 16th transmission round, packet loss is recognized by a triple duplicate ACK. If there was a timeout, the congestion window size would have dropped to 1.

f) After the 22nd transmission round, segment loss is detected due to timeout, and hence the congestion window size is set to 1.

g) TCP congestion advoidance is operating in the intervals [6,16] and [17,22]

h) The threshold is initially 32, since it is at this window size that slowtart stops and congestion avoidance begins.

i) During the 1st transmission round, packet 1 is sent; packet 2-3 are sent in the 2nd transmission round; packets 4-7 are sent in the 3rd transmission round; packets 8-15 are sent in the 4th transmission round; packets15-31 are sent in the 5th transmission round; packets 32-63 are sent in the 6th transmission round; packets 64 – 96 are sent in the 7th transmission round. Thus packet 70 is sent in the 7th transmission round.

CH4

8.

a)

Prefix Match	Link Interface
11100000	0
11100001 00000000	1
11100001	2
otherwise	3

b) Prefix match for first address is 4th entry: link interface 3
Prefix match for second address is 2nd entry: link interface 1
Prefix match for first address is 3rd entry: link interface 2

10.

223.1.17.0/25
223.1.17.128/26
223.1.17.192/26

14.

Destination Address Range	Link Interface
00000000 through 00111111	0
01000000 through 01111111	1
10000000 through 10111111	2
11000000 through 11111111	3

number of addresses in each range = $2^6 = 64$

20.

a. $D_x(w) = 2$, $D_x(y) = 4$, $D_x(u) = 7$

b.

First consider what happens if c(x,y) changes. If c(x,y) becomes larger or smaller (as long as c(x,y) >=1) , the least cost path from x to u will still have cost at least 7. Thus a change in c(x,y) (if c(x,y)>=1) will not cause x to inform its neighbors of any changes. If $c(x,y)= \delta<1$, then the least cost path now passes through y and has cost $\delta+6$.

Now consider if c(x,w) changes. If $c(x,w) = \varepsilon \leq 1$, then the least-cost path to u continues to pass through w and its cost changes to $5 + \varepsilon$; x will inform its neighbors of this new cost. If $c(x,w) = \delta > 6$, then the least cost path now passes through y and has cost 11; again x will inform its neighbors of this new cost.

c. Any change in link cost c(x,y) (and as long as c(x,y) >=1) will not cause x to inform its neighbors of a new minimum-cost path to u .

23.

		Cost to				
		u	v	x	y	z
	v	∞	∞	∞	∞	∞
From	x	∞	∞	∞	∞	∞
	z	∞	6	2	∞	0

		Cost to				
		u	v	x	y	z
	v	1	0	3	∞	6
From	x	∞	3	0	3	2
	z	7	5	2	5	0

		Cost to				
		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

		Cost to				
		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

25.

Step	*N'*	*D(t),p(t)*	*D(u),p(u)*	*D(v),p(v)*	*D(w),p(w)*	*D(y),p(y)*	*D(z),p(z)*
0	x	∞	∞	3,x	6,x	6,x	8,x
1	xv	7,v	6,v	3,x	6,x	6,x	8,x
2	xvu	7,v	6,v	3,x	6,x	6,x	8,x
3	xvuw	7,v	6,v	3,x	6,x	6,x	8,x
4	xvuwy	7,v	6,v	3,x	6,x	6,x	8,x
5	xvuwyt	7,v	6,v	3,x	6,x	6,x	8,x
6	xvuwytz	7,v	6,v	3,x	6,x	6,x	8,x

26.

Node x table

Cost to

From		x	y	z
	x	0	5	2
	y	∞	∞	∞
	z	∞	∞	∞

Cost to

From		x	y	z
	x	0	5	2
	y	5	0	6
	z	2	6	0

Node y table

Cost to

From		x	y	z
	x	∞	∞	∞
	y	5	0	6
	z	∞	∞	∞

Cost to

From		x	y	z
	x	0	5	2
	y	5	0	6
	z	2	6	0

Node z table

Cost to

From		x	y	z
	x	∞	∞	∞
	y	∞	∞	∞
	z	2	6	0

Cost to

From		x	y	z
	x	0	5	2
	y	5	0	6
	z	2	6	0