CS 408 – HW 4 – Oytun Kuday Duran 28357

1) (10 points) A certain router receives a datagram of size 7000 B (including the header). However, all of its output ports have an MTU (maximum transfer unit) of 1500 B, thus the original datagram should be fragmented into several smaller datagrams when forwarded. For each of the output fragments write the following fields of the IP header: length, ID, flag and offset! Assume 20 B for the IP header.

Ans:
(id is same with unfragmanted version of datagram lets call x)

Fragment 1: 20 header + 1480 payload (id =x, Length: 1500 offset: 0 , fragflag=1)

Fragment 2: 20 header + 1480 payload (id =x, Length: 1500 offset: 1480/8 =185, fragflag=1)

Fragment 3: 20 header + 1480 payload (id =x, Length: 1500 offset: 2960/8 = 370, fragflag=1)

Fragment 4: 20 header + 1480 payload (id =x, Length: 1500 offset: 4440/8 = 555, fragflag=1)

Fragment 5: 20 header + 1060 payload (id =x, Length: 1080 offset: 5920/8 = 740, fragflag=0)

2) (10 points) A certain autonomous system (AS) is assigned a subnet which to the outside world (Internet) is advertised as 24.23.5.0 /24. Each router has its number of hosts attached to it, which includes the router interface on that particular side of the subnet. As a network administrator, your job is to properly and efficiently do the subnetting inside this AS! (Note: for subnets with 2 interfaces we suggest to also include the network and broadcast domain) (Hint: start with the largest subnets first)

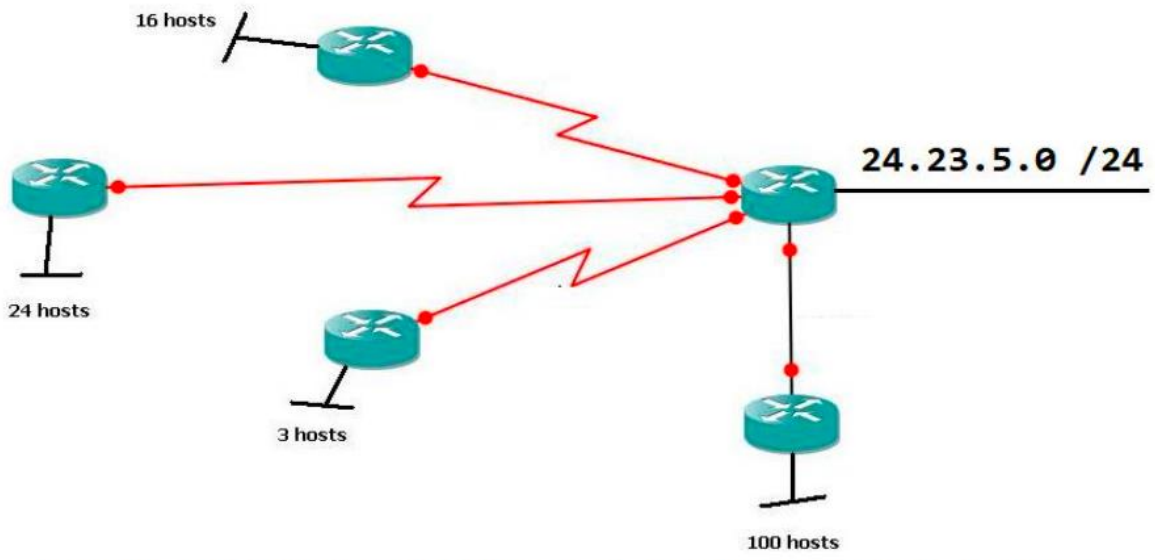


Fig.1. An autonomous system (AS) with an assigned subnet of 24.23.5.0 /24.

Fig.1. An autonomous system (AS) with an assigned subnet of 24.23.5.0 /24.

ANS:

24.23.5.0/24 is

00011000.00010111.00000101.00000000

Last 8 bits are free.

We need to have at least hosts + 4 possible combinations as we discussed in W14-L1 since they have 2 for interfaces, a network and a broadcast ip.

[4 subnets could also be represented using 2 bits (00,01,10,11) if they had equal hosts]

For 100 hosts we need 2^7 => 100+4 <128 7 bits to represent

So it can be 24.23.5.0 to 24.23.5.103 and 103 is for broadcast (network ip is 24.23.5.0.)

24.23.5.0/25

For 24 hosts we need 2^5 => 24+4 < 32 5 bits to represent

So it can be 24.23.5.104 to 24.23.5.131 and 131 is for broadcast (network ip is 24.23.5.104.)

24.23.5.104/27

For 16 hosts we need 2^5 => 16+4 < 32 5 bits to represent.

So it can be 24.23.5.132 to 24.23.5.151 and 151 is for broadcast (network ip is 24.23.5.132.)

24.23.5.132/27

For 3 hosts we need 2^3 => 3+4 <8 3 bits to represent

So it can be 24.23.5.152 to 24.23.5.158 and 158 is for broadcast ((network ip is 24.23.5.152.)

24.23.5.152/29

In total we need 200 bits < 256 so 8 bits will be enough

3) (10 points) Inside your AS you have the 6 subnets given bellow. Summarize them so when advertised to the outside world you'll have as least entries (subnets) to advertise as possible. Of course, avoid over-summarization.

ANS:

243.157.50.0 /22 => 11110011.10011101.001100**10**.00000000
243.157.54.0 /22 => 11110011.10011101.001101**10**.00000000
243.157.58.0 /22 => 11110011.10011101.001110**10**.00000000
243.157.62.0 /22 => 11110011.10011101.001111**10**.00000000
243.157.10.0 /22 => 11110011.10011101.000010**10**.00000000
243.157.42.0 /22 => 11110011.10011101.001010**10**.00000000

As you can see, first 18 bits are on all are the same, if we put rest 0,
11110011.10011101.00000000.00000000 => 243.157.0.0/18

4) (10 points) In Fig.2 you have the topology of a certain network. Construct the forwarding table for router u towards all the other routers using the Dijkstra's algorithm. Show all of the steps!

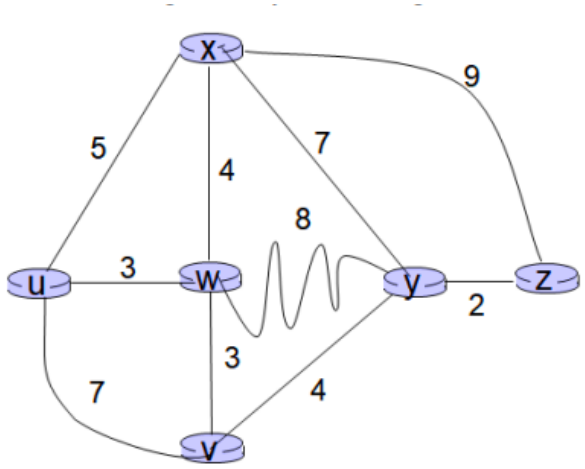
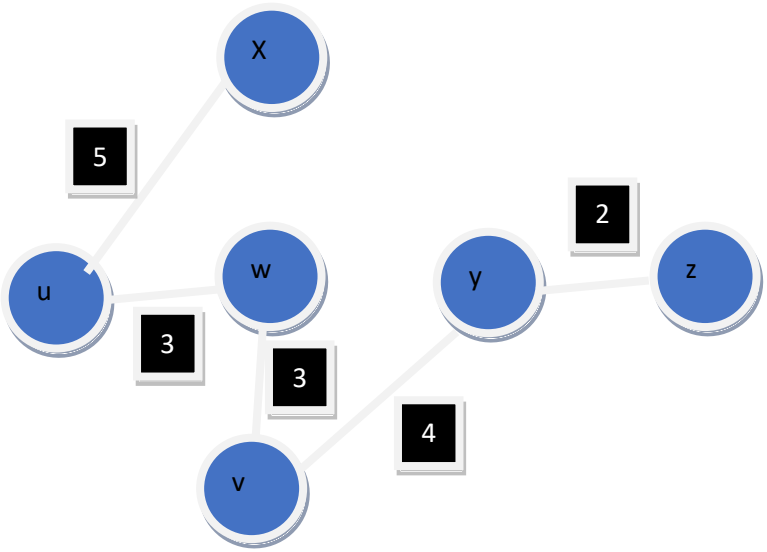


Fig.2. Topology of a certain network

5)

ANS:

Iteration	x	y	z	u	v	w
1	∞,-	∞,-	∞,-	0,-	∞,-	∞,-
2	5,u	∞,-	∞,-	0,-	7,u	3,u
3	5,u	11,w	∞,-	0,-	6,w	3,u
4	5,u	11,w	14,x	0,-	6,w	3,u
5	5,u	10,v	14,x	0,-	6,w	3,u
6	5,u	10,v	12,y	0,-	6,w	3,u



<= GRAPH

5) (15 points) In Fig.3 you have the topology of a certain network. Construct the forwarding table for all of the routers towards all the other routers in the network using the Bellman-Ford's algorithm. You can assume that all the routers send their newly computed/changed distance vectors (DV's) values to their neighbor at the same time, i.e. for t=0 all of them have only the DVs for their neighbors, at t=1 they do the first exchange of DVs, at t=2 they do the second exchange of DVs, etc. Show all of the steps until the algorithm converges for all of the routers (i.e. there are no more changes in DVs)! (Note: if everything as done correctly, you shouldn't have more to do more than 3 iterations (including the initialization) for any of the routers.) (Hint: fill-up the all of the tables below corresponding to new DV values after an exchange with the neighbors. The initial tables for all of the routers are given.)

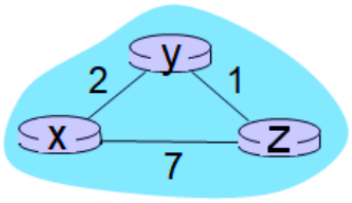


Fig.3. Topology of a certain network

t=0

Dx()	X	Y	Z
X	0	2	7
Y	∞	∞	∞
Z	∞	∞	∞

DV table of X

Dy()	X	Y	Z
X	∞	∞	∞
Y	2	0	1
Z	∞	∞	∞

DV table of Y

Dz()	X	Y	Z
X	∞	∞	∞
Y	∞	∞	∞
Z	7	1	0

DV table of Z

t=1

Dx()	X	Y	Z
X			
Y			
Z			

DV table of X

Dy()	X	Y	Z
X			
Y			
Z			

DV table of Y

Dz()	X	Y	Z
X			
Y			
Z			

DV table of Z

t=2

Dx()	X	Y	Z
X			
Y			
Z			

DV table of X

Dy()	X	Y	Z
X			
Y			
Z			

DV table of Y

Dz()	X	Y	Z
X			
Y			
Z			

DV table of Z

ANS: Using Bellman-Ford Equations: $D_x(y) = \min \{ c_{x,y} + D_y(y) , c_{x,z} + D_z(y) \}$ *and*

$D_x(z) = \min \{ c_{x,y} + D_y(z) , c_{x,z} + D_z(z) \}$

$D_y(z) = \min \{ c_{z,y} + D_x(z) , c_{y,x} + D_z(z) \}$

Although it may not look very clear because of my bad copy-paste skills and limited time, I wanted to explain that we basically take minimums of cost+path's of different vertices and update at every iteration. We need to take a look at previous graph to get D function values to consider next values in tables. In addition, minimum of something and infinity is something itself (The values doesn't change in first iterations is because of this.).

Initialization is as given in figure step t=0,

Lets look for t=1

Dx()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,3)=3
Y	Min(2,3)=2	Min(0,sth) = 0	Min(1,9)=1
Z	7	1	Min(0,sth) = 0

DV table of X

Dy()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,inf)=7
Y	Min(2,3)=2	Min(0,sth) = 0	Min(1,9)=1
Z	7	1	Min(0,sth) = 0

DV table of Y

Dz()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,inf)=7
Y	Min(2,3)=2	Min(0,sth) = 0	1
Z	Min(7,3)=3	Min(1,9)=1	Min(0,sth) = 0

DV table of Z

Lets look for t=2

Dx()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,3)=3
Y	Min(2,3)=2	Min(0,sth) = 0	Min(1,9)=1
Z	Min(7,3)=3	Min(1,9)=1	Min(0,sth) = 0

DV table of X

Dy()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,3)=3
Y	Min(2,3)=2	Min(0,sth) = 0	Min(1,9)=1
Z	Min(7,3)=3	Min(1,9)=1	Min(0,sth) = 0

DV table of Y

Dz()	X	Y	Z
X	Min(0,sth) = 0	Min(2,3)=2	Min(7,3)=3
Y	Min(2,3)=2	Min(0,sth) = 0	Min(1,9)=1
Z	Min(7,3)=3	Min(1,9)=1	Min(0,sth) = 0

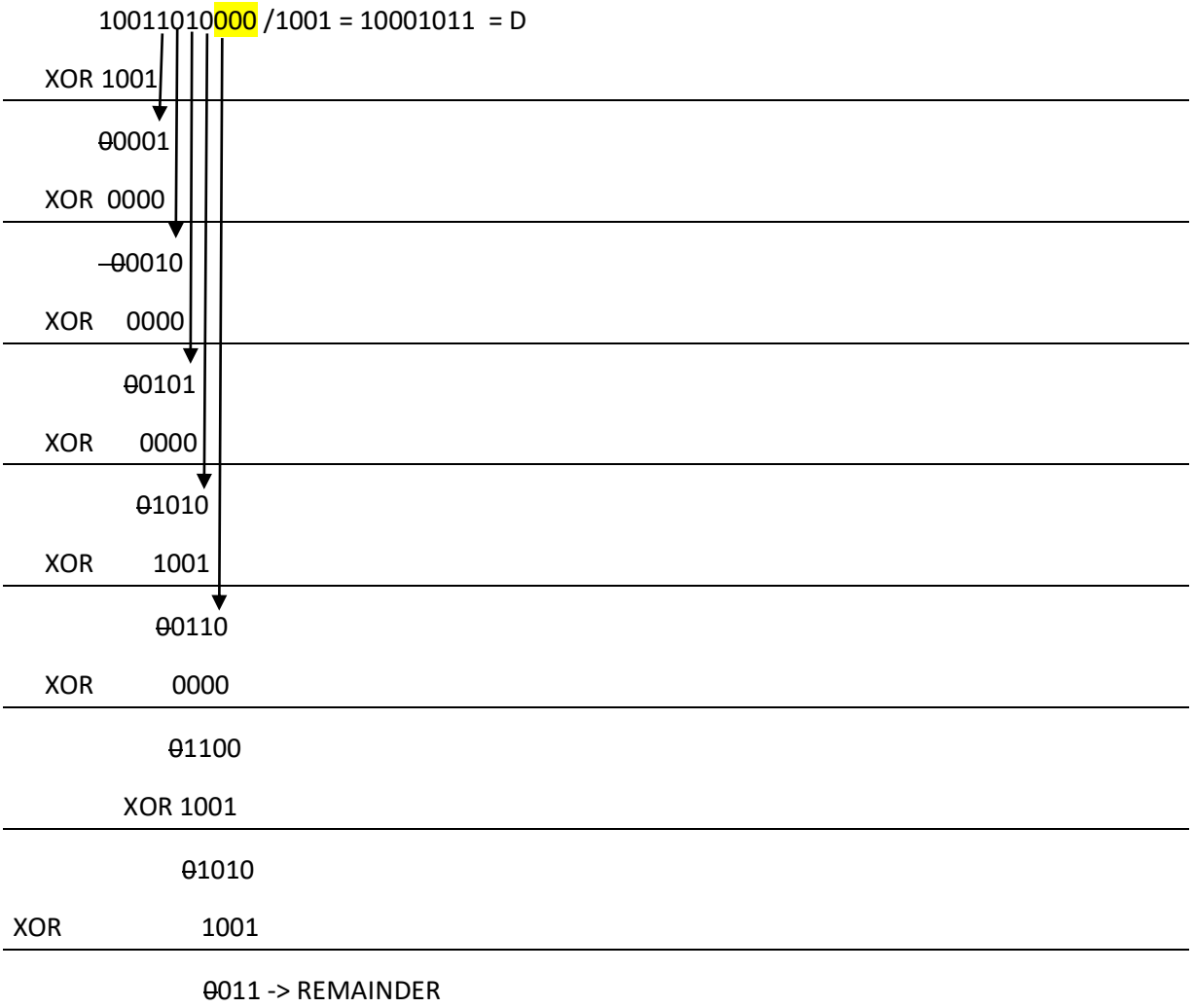
DV table of Z

6) (10 points) A sender has this data payload to send D=10011010. If both the sender and the receiver have agreed to use 1 + x 3 as their generator (i.e. G=1001), then compute the CRC (Cyclic Redundancy Check) for D! Show all the details!

ANS:

R+1 is 4 bits so r is 3 bits.

Add 3 bits to end of D



D' = D|R = 10001011 | 011 = 10001011011

7) (15 points)

a) (5 points) A sender has this data payload to send D=01011010001011. It uses a 2D parity check with 5 rows and 4 columns and odd parity (i.e. in each row and column, including the parity bits, the number of 1s should be an odd number). Construct the message that will be send after 2D parity is added and show it in matrix form. (Note: If the number of bits in the payload D is not enough to fill-up the matrix with 5 rows and 4 columns, at the end of D you can append the necessary number of zeros so as to fill-up the matrix)

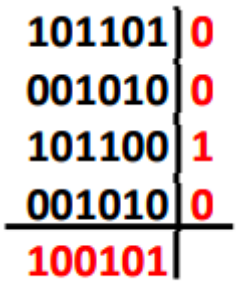
ANS:

0	1	0	1	1
1	0	1	0	1
0	0	1	0	0
1	1	0	0	1
0	0	0	0	1
1	1	1	0	

01011 | 10101 | 00100 | 11001 | 00001| 11100 = 01011 10101 00100 11001 00001 11100

b) (10 points) In the table below you have the message that a certain sender which uses 2D even parity check for error detection and/or correction has sent. Are there any transmission errors? If so, where? If so, can you correct them? Why?

ANS:



1	0	1	1	0	1	0
0	0	1	0	1	0	0
1	0	1	1	0	0	1
0	0	1	0	1	0	0
1	0	0	1	0	1	

Yes there are errors, odd 1's in some columns. To correct errors we can change 1's to 0's in first row and it would still have even parity but the errors could be corrected. There could be other possible solutions too. Since all row's and cols have even parity, the problem would be solved. (count of 1's would go 2 from 4 as i showed above when we correct it. 2 1's to 0's.)

0	0	1	0	0	1	0
0	0	1	0	1	0	0

1	0	1	1	0	0	1
0	0	1	0	1	0	0
1	0	0	1	0	1	

8) (15 points) A sender sends an n bit message through an unreliable channel which has a bit-error probability p. What is the probability of:

a) (5 points) exactly one bit is flipped from the transmitted message

All are not flipped except 1 * 1 flipped

$P(C1) = p^1(1-p)^{n-1}$

b) (5 points) at least one bit is flipped from the transmitted message

All cases – case where no bit flipped:

$P(C2) = 1- (1-p)^n$

c) (5 points) exactly b bits are flipped from the transmitted message

$P(C3) = p^b(1-p)^{n-b}$

9) (25 points) Consider the figure below (Fig.4), which shows the arrival of 6 messages for transmission at different multiple access wireless nodes at times t = <0.8, 1.2, 2.9, 3.1, 4.3, 4.6> and each transmission requires exactly one-time unit.

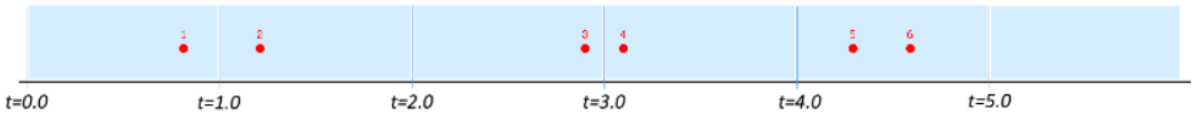


Fig.4. Transmission times of different multiple access nodes sharing a transmission medium

Fig.4. Transmission times of different multiple access nodes sharing a transmission medium

a) (5 points) Aloha:

unslotted Aloha: simpler, no synchronization

collision probability increases with no synchronization: frame sent at t_0 collides with other frames sent in $[t_0-1,t_0+1]$

a.1) Suppose all nodes are implementing the Aloha protocol. For each message, indicate the time at which each transmission begins. Separate each value with a comma and no spaces.

Node 1 will go from 0.8 to 1.8

Node 2 will go from 1.2 to 2.2
Node 3 will go from 2.9 to 3.9
Node 4 will go from 3.1 to 4.1
Node 5 will go from 4.3 to 5.3
Node 6 will go from 4.6 to 5.6

a.2) Which messages transmit successfully? Write your answer as a comma separated list with no spaces using the messages' numbers

I believe none of the messages transmit successfully as every node's transaction intersect with one another.

b) (5 points) Slotted Aloha
b.1) Suppose all nodes are implementing the Slotted Aloha protocol. For each message, indicate the time at which each transmission begins. Separate each value with a comma and no spaces.
1.0,2.0,3.0,4.0,5.0,6.0
or
1.0,2.0,3.0,4.0,5.0,5.0
I couldn't be sure since last 2 packages are in conflict and one of them needs to be retransmitted.

b.2) Which messages transmit successfully? Write your answer as a comma separated list with no spaces using the messages' numbers

1,2,3,4

c) (5 points) CSMA
c.1) Suppose all nodes are implementing Carrier Sense Multiple Access (CSMA), but without collision detection. Suppose that the time from when a message transmission begins until it is beginning to be received at other nodes is 0.4 time units. (Thus if a node begins transmitting a message at t=2.0 and transmits that message until t=3.0, then any node performing carrier sensing in the interval [2.4, 3.4] will sense the channel busy.) For each message, indicate the time at which each message transmission begins, or indicate that message transmission does not begin due to a channel that is sensed busy when that message arrives. Separate each value with a comma and no spaces, and if the channel is sensed busy, substitute it with 's'

N1 -> 0.8+0.4 to 1.8+0.4 = 2.2, Channel is busy between 1.2 – 2.2
N2-> 1.8 + 0.4 = 2.2 to 2.8 + 4 = 3.2. Since channel is busy at 2.2, it won't transmit.
N3-> 2.9+ 0.4 to 3.9+0.4 = 3.3 to 4.3. Channel is free so it can transmit.
N4-> 3.1 + 0.4 to 4.1 + 0.4 = 3.5 to 4.5, channel is busy so wont transmit.
N5 4.3 + 0.4 to 5.3 + 0.4 = 4.7 to 5.7, channel is free so it will transmit.
N6 4.6+0.4 to 5.6 +0.4 = 5.0 to 6.0, Channel is busy so wont transmit.

1.2,s,3.3,s,4.7,s

c.2) Which messages transmitted successfully? Write your answer as a comma separated list with no spaces using the messages' numbers

1,3,5

d.1) Suppose all nodes are implementing Carrier Sense Multiple Access (CSMA), with collision detection (CSMA/CD). Suppose that the time from when a message transmission begins until it is beginning to be received at other nodes is 0.4 time units, and assume that a node can stop transmission instantaneously when a message collision is detected. (Thus if a node begins transmitting a message at t=2.0 and transmits that message until t=3.0, then any node performing carrier sensing in the interval [2.4, 3.4] will sense the channel busy.) For each message, indicate the time at which each message transmission begins, or indicate that message transmission does not begin due to a channel that is sensed busy when that message arrives. Separate each value with a comma and no spaces, and if the channel is sensed busy, substitute it with 's

N1 -> 0.8+0.4 to 1.8+0.4 = 2.2, Channel is busy between 1.2 – 2.2
N2-> 1.8 + 0.4 = 2.2 to 2.8 + 4 = 3.2. Since channel is busy at 2.2, it won't transmit.
N3-> 2.9+ 0.4 to 3.9+0.4 = 3.3 to 4.3. Channel is free so it can transmit.
N4-> 3.1 + 0.4 to 4.1 + 0.4 = 3.5 to 4.5, channel is busy so wont transmit.
N5 4.3 + 0.4 to 5.3 + 0.4 = 4.7 to 5.7, channel is free so it will transmit.
N6 4.6+0.4 to 5.6 +0.4 = 5.0 to 6.0, Channel is busy so wont transmit.

1.2,s,3.3,s,4.7,s

q

d.2) Which messages transmitted successfully? Write your answer as a comma separated list with no spaces using the messages' numbers

1,3,5

d.3) At what time did each message stop transmitting due to a collision. Write your answer as a comma separated list with no spaces using the messages' numbers in order, and if a message didn't stop, write 'x' for that message

x,2.2,x,3.5,x,5.0