CS/ECE 438: Communication Networks **Spring 2023**

Homework 3

Handed Out: March 26th, 2023 *Due: 11:59pm, April 2nd, 2023*

TA: Federico Cifuentes-Urtubey

- Homework assignments must be submitted online through GradeScope. Hard copies are not accepted. Please submit a pdf file to GradeScope (`https://www.gradescope.com`). You can either type your solution or scan a **legible** hand-written copy. We will not correct anything we do not understand. Contact the TAs via Campuswire if you face technical difficulties in submitting the assignment. Note: Please select your page(s) for each question during submission on GradeScope; Otherwise, your submission will not be graded.
- Homework assignments can be done in groups of two, but only one person needs to submit on GradeScope. Remember to include your partners name in the solution and on GradeScope by editing "Group Members". It is your responsibility that your partner's name is included. For detailed instruction please refer to (`https://youtu.be/rue7p_kATLA`).
- You can use Campuswire to find a partner. We highly recommend working in groups. You will not get extra credit for working alone.
- Please use Campuswire and come to office hours if you have questions about the homework. Failure to understand the solutions will be the student's fault.
- While we encourage discussion within and outside of the class, cheating and copying is strictly prohibited. Copied solutions will result in the entire assignment being discarded from grading at the very least and a report filed in the FAIR system. It is also your responsibility to ensure that your partner obeys the academic integrity rules as well.

1 TCP RTT Estimation – 7 points

One difficulty with the original TCP SRTT estimator is the choice of an initial value. In the absence of any special knowledge of network conditions, the typical approach is to pick an arbitrary value, such as 3 seconds, and hope this will converge quickly to an accurate value. If this estimate is too small, TCP will perform unnecessary retransmissions. If it is too large, TCP will wait a long time before retransmitting if the first segment is lost. Also, the convergence might be slow.

1. Choose $\alpha = 0.7$ and RTT-timeout(0) = 1 seconds, and assume all measured RTT values = 0.5 second with no packet loss. What is RTT-timeout(20)? Recall,

$$\text{RTT-timeout}(k+1) = \alpha \times \text{RTT-timeout}(k) + (1-\alpha) \times \text{RTT}(k+1)$$

   Describe your solution approach AND provide the numerical result (approximate to 4th decimal place).

2. Using the same values as in above part, what happens if we use $\alpha = 0.5$ or $\alpha = 0.95$? Provide a numerical result for RTT-timeout(20) in both cases, then describe the effect of a larger or smaller α on the RTT estimation procedure.

Solution: (7 = 3+4 points)

1. Based on the formula and $\alpha = 0.7$, we can calculate the n^{th} RTT-timeout value using:

$$\text{RTT-timeout}(n) = \alpha^n \times \text{RTT-timeout}(0) + (1-\alpha)\ \alpha^n \times \sum_{j=1}^{n} \frac{\text{RTT}(j)}{\alpha^j}$$

   We are given that all measured RTT values are 0.5 sec. Hence we can rewrite above equation as

$$\text{RTT-timeout}(n) = \alpha^n \times \text{RTT-timeout}(0) + 0.5 \times (1-\alpha)\ \alpha^n \times \sum_{j=1}^{n} \frac{1}{\alpha^j}$$

   By using initial conditions, we can calculate RTT-timeout(20) as **0.5004**.

2. With $\alpha = 0.5$, we have RTT-timeout(20) $= 0.5000$

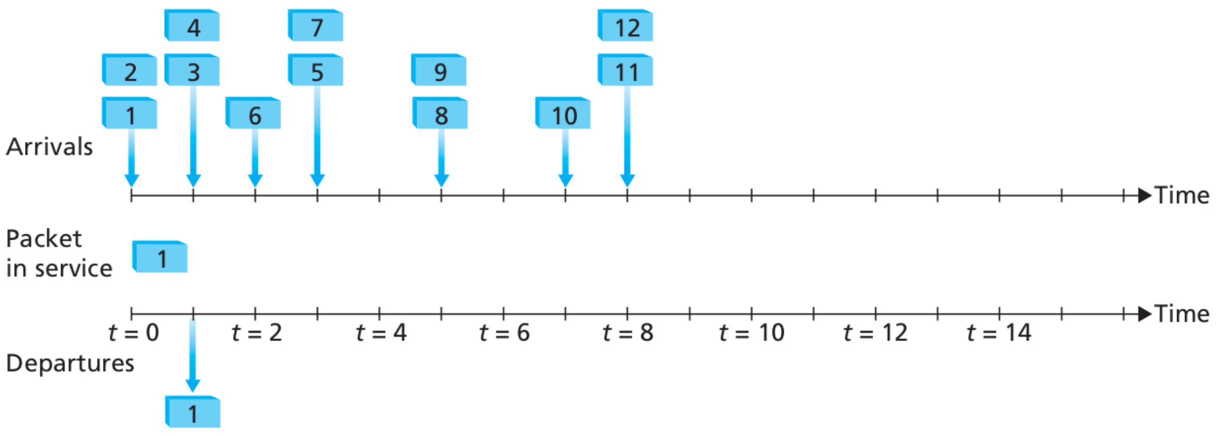
   With $\alpha = 0.95$, we have RTT-timeout(20) $= 0.6792$

   A large α will give more weight to the initial estimated value of RTT-timeout, hence as we can see in the case of $\alpha = 0.95$, the convergence of RTT-timeout to the true RTT value is slower. With a small α, more weight will be given to the new measured RTTs, and hence convergence to the true RTT is faster. This is evident from the calculations for the case of $\alpha = 0.5$.

2 Fair Queuing – 18 points

Consider Fig. 1. Now consider the following scenarios:

Figure 1:



- Packets are scheduled using FIFO policy
- Packets are scheduled using Highest Priority First policy. Assume odd-numbered packets are high priority and even-numbered packets are low priority.
- Packets are scheduled using Round Robin policy. Assume that packets (1, 4, 6, 7, 8, 9, 10) are from class 1, and packets (2, 3, 5, 11, 12) are from class 2. Scheduling starts with class 1 at $t = 0$.
- Packets are scheduled using Weighted Fair Queuing (WFQ) policy. Assume there are three classes. Let us denote the packet ID of packet i as ID_i. Class j (where $j = 0, 1, 2$), will contain packets which satisfy $\{i | ID_i \% 3 = j\}$. Let the three classes (i.e. Class 0,1 and 2) have the weights 3, 2 and 1 respectively. Scheduling starts with class 0 at $t = 0$.

Note: In case of ties under the above schemes, you must resort to FIFO scheme. In Fig. 1, you can assume that Packet 1 arrives before Packet 2 at $t = 0$, Packet 3 before Packet 4 at $t = 1$ and so on. Also assume that packets can be immediately scheduled for transmission when they arrive. For Round Robin and WFQ, you must skip a Class if packets are not available for that particular class.

1. Fill this Table. 1 for which packet departs at each time point and the delay of each packet (i.e., delay is time interval between the arrival and departure of a packet).
2. What is average delay for these 4 polices? For Highest priority, Round Robin, and WFQ, you must also list average delay for each class separately. (In WFQ, the average delay among different classes might not be perfect in this question.)
3. What observations can you draw about the average delay from the above 4 polices? Two concise observations are sufficient.

Table 1: Packet Scheduling

Time of Departure (t in sec)	FIFO		Highest Priority		Round Robin		WFQ	
	Packet	Delay	Packet	Delay	Packet	Delay	Packet	Delay
1	1	1	1	1	1	1	1	1
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								

Solution: (18 = 12 + 4 + 2 points)

1. See table below

Time of Departure (t in sec)	FIFO		Highest Priority		Round Robin		WFQ	
	Packet	Delay	Packet	Delay	Packet	Delay	Packet	Delay
1	1	1	1	1	1	1	1	1
2	2	2	3	1	2	2	4	1
3	3	2	2	3	4	2	2	3
4	4	3	5	1	3	3	3	3
5	6	3	7	2	6	3	6	3
6	5	3	9	1	5	3	9	1
7	7	4	4	6	7	4	7	4
8	8	3	6	6	8	3	10	1
9	9	4	11	1	11	1	5	6
10	10	3	8	5	9	5	12	2
11	11	3	10	4	12	3	8	6
12	12	4	12	4	10	5	11	4

2.
   - Average Delay in case of FIFO is: 2.91
   - Average Delay in case of Highest Priority is: 2.91

     Average Delay for Class High Priority: 1.2

     Average Delay for Class Low Priority: 4.7
   - Average Delay in case of Round Robin is: 2.91

     Average Delay for Class 1 = (1+2+3+4+3+5+5)/7 = 3.28

Average Delay for Class 2 = (2+3+3+1+3)/5 = 2.4

- Average Delay in case of WFQ is: 2.91

  Average Delay for Class 0: 2.25

  Average Delay for Class 1: 1.75

  Average Delay for Class 2: 4.75

3. **Observations:**

   - The average delay in all cases is the same. This is expected because in each case, the amount of time required to push all packets out of the queue will be the same.
   - For Round Robin we can see that the delay for each class is close, whereas in WFQ we can see that the class with least weight has maximum delay. In Highest Priority, the class with Higher Priority has significantly smaller delay than the low priority class.

3 Forwarding and CIDR – 15 points

1. Consider a router that interconnects three subnets: Subnet A, Subnet B and Subnet C. Suppose all of the interfaces in each of these three subnets are required to have the prefix 200.20.15.0/24. Also suppose that Subnet A is required to support up to 80 interfaces, and Subnets B and C are each required to support up to 25 interfaces. Provide three network addresses (of the form a.b.c.d/x) for each of the above subnet that satisfy these constraints.

Suppose a router has built up the routing table shown below in Table. 2. CIDR addresses are used, with "/22" indicating a mask of 22 1's followed by 10 0's.

2. How many individual IP addresses match each Net/Masklength pair? (Compute this for all entries in the table except the last one, i.e. for the default Masklength entry).
3. The router can deliver packets directly over interfaces 1, 2, 3, 4, or it can forward to routers R1, R2, R3. Specify the next hop for each of the following destinations. Use the longest prefix match, i.e., if a destination matches more than one line of the table, the longest match is used.

Net/Masklength	NextHop
128.174.240.0/20	Interface 1
128.174.240.128/25	R1
128.174.240.17	R2
128.174.252.0/22	Interface 3
128.174.240.16/29	R3
128.174.248.0/22	Interface 2
default	Interface 4

Table 2:

(a) 128.174.240.17

(b) 128.174.245.17

(c) 128.174.250.17

(d) 128.174.254.17

(e) 128.174.225.17

(f) 128.174.240.18

Solution: (15 = 3+6+6 points)

1. For subnet A we have 200.20.15.128/25
   For subnet B we have 200.20.15.64/27
   For subnet C we have 200.20.15.96/27

   OR

   For subnet A we have 200.20.15.0/25
   For subnet B we have 200.20.15.192/27
   For subnet C we have 200.20.15.128/27

2. - 128.174.240.0/20 - 4096 IPs
   - 128.174.240.128/25 - 128 IPs
   - 128.174.240.17 - 1 IP
   - 128.174.252.0/22 - 1024 IPs
   - 128.174.240.16/29 - 8 IPs
   - 128.174.248.0/22 - 1024 IPs

3. (a) 128.174.240.17 - R2

   (b) 128.174.245.17 - Interface 1

   (c) 128.174.250.17 - Interface 2

(d) 128.174.254.17 - Interface 3

(e) 128.174.225.17 - Interface 4

(f) 128.174.240.18 - R3

4 DHCP and NAT – 8 points

Alice and Bob are neighbors and they each buy a new home wireless router. After connecting each of their laptops to their own router they each enter the command `hostname -i`, which prints out their IP address.

1. Briefly describe the process how they get the IP address.
2. After doing this, is it possible that they get the same IP address, if the routers use private subnet addresses?
3. Give one reason that wide-spread deployment of IPv6 would let them get rid of the NAT service provided by thier routers.
4. Give one reason that they might want to continue using the router's NAT service even if they could use IPv6.

Solution: 8 = 2+2+2+2 points

1. They obtain their IP address using the DHCP protocol. This stands for Dynamic Host Configuration Protocol. In this protocol, each host trying to enter a network first needs an IP address for itself and it obtains this IP address using the DHCP protocol. The host first sends out a broadcast message which is a "DHCP discover" message. It has to do this since it does not know the address of the DHCP server. It broadcasts a UDP packet, with destination IP address: 255.255.255.255 and source IP address being 0.0.0.0. A DHCP server if present, responds with a DHCP "offer message". In this message, the DHCP server suggests a potential IP address the host could use. After receiving this message the host sends back a "DHCP request" message confirming that he will use the IP address. Finally the host received a "DHCP ACK" from the DHCP server to complete the process of assigning an IP address for the client. When Alice and Bob got their IP address this is the process they went through and

their host communicated with a DHCP server in order to get the IP address.

2. This is possible because this address is a private address and is not visible to the outside world. Also since Alice and Bob are not part of the same local network, they can have the same IP address. The NAT present in each of their routers gives them a different effective IP address, and this is essentially the IP address that Alice and Bob use for communicating with the outside world. Hence as far as the outside world is concerned, the IP address of Alice and Bob are actually different.

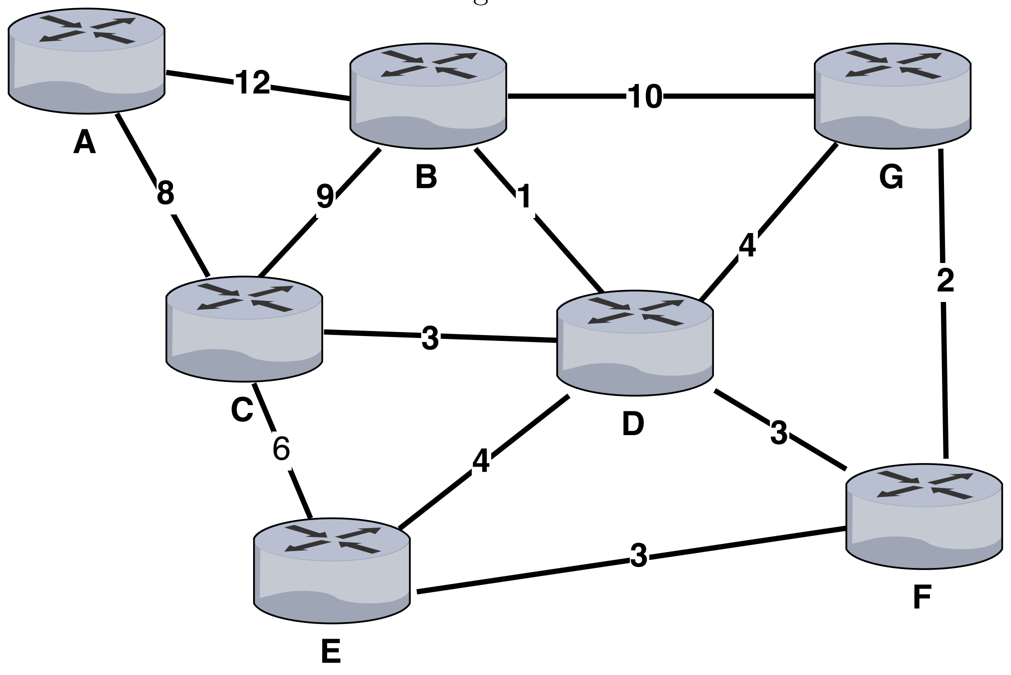
3. The primary reason for using NAT based routers is to overcome the severe addressing shortage faced today. IPv4 uses 32 bit IP addresses which are getting allocated very fast. However with IPv6 we now have 128 bits for addresses which is a lot of IP addresses. Hence we do not need to assign multiple devices the same IP address now using the NAT technique, and in fact every device now could be assigned its own IP address. Hence wide-spread deployment of IPv6 can get rid of the entire NAT scheme.

4. We might want to continue using NAT since they provide a security advantage. The real IP address of the clients are not visible to the outside world and hence the IP address of the client remains hidden from malicious attackers. The IP address visible to the outside world is actually the IP address of the router (which happens due to NAT), and hence in this way the IP address of the host is safeguarded.

5 Distance Vector Routing – 21 points

Consider the following network topology as shown in Fig. 1.

Figure 2:



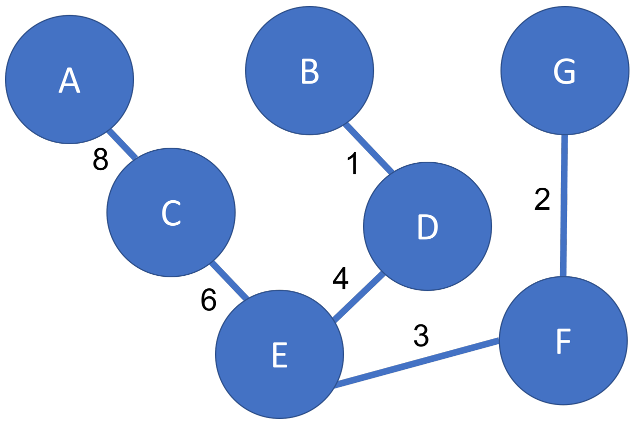
1. Derive and draw the shortest-path tree from nodes E using the Dijkstra's algorithm. Show how the cost of path and predecessor node along path converge step-by-step in a table as demonstrated in the lecture slides.
2. Compute routing tables for nodes B, D and E using the distance vector algorithm. The table rows should include a *Destination*, *Distance*, and *Next Hop*. Please show intermediate results for partial credit.
3. Now consider a situation in which link between the D-C becomes congested and its costs become 30. List a sequence of updates to routing tables at nodes B and D, in order, until the routing tables converge.

Solution: (21 = 3 + 9 + 9 points)

1. Computing the spanning tree (shortest-path tree) for E:

Steps	N'	D(A) P(A)	D(B) P(B)	D(C) P(C)	D(D) P(D)	D(F) P(F)	D(G) P(G)
0	E	INF	INF	6, E	4, E	3, E	INF
1	E,F	INF	INF	6, E	4, E		5, F
2	E,F,D	INF	5, D	6, E			5, F
3	E,F,D,B	17, B		6, E			5, F
4	E,F,D,B,G	17, B		6, E			
5	E,F,D,B,G,C	14, C					
6	E,F,D,B,G,C,A						

Spanning tree for node E:



2. With the Distance Vector algorithm, each iteration updates routing tables for each node as it receives new information from its neighbors. A routing table update can also occur if there is a local change detected by the node in its own associated link costs. Here, however, we are not assuming such changes and hence a routing table is updated only if the node gets an updated routing table from a neighbor. When the algorithm stars, the routing table initializes as below:

	Node A; NextHop	Node B; NextHop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node A	0; A	12; B	8; C	INF	INF	INF	INF
Node B	12; A	0; B	9; C	1; D	INF	INF	10;G
Node C	8; A	9; B	0; C	3; D	6; E	INF	INF
Node D	INF	1; B	3; C	0; D	4; E	3; F	4; G
Node E	INF	INF	6; C	4; D	0; E	3; F	INF
Node F	INF	INF	INF	3; D	3; E	0; F	2; G
Node G	INF	10; B	INF	4; D	INF	2; F	0; G

Rows correspond to the source node, and columns correspond to the destination node. In each column, both the destination and next hop are listed. In each cell of the matrix, we have two entries: a number followed by a node. This number is the distance between the source node (row) and the destination node (column), and the node listed in the cell is the next hop node from the source node.

When a node detects a change in its own routing table, it will forward its routing table to its neighbors, who in turn update their own routing table to accommodate for changes in its neighbor's routing table. To make this problem concise, we will assume synchronous behavior for routing table passing and updating. (This won't affect the convergence.) We can also assume all nodes pass their routing table to their neighbors at the same instant, and that all nodes update their own routing table at the same next instant.

In the next iteration of the algorithm, we have the following updated table:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node A	0; A	12; B	8; C	11; C	14; C	INF	22; B
Node B	12; A	0; B	4; D	1; D	5; D	4; D	5; D
Node C	8; A	4; D	0; C	3; D	6; E	6; D	7; D
Node D	11; C	1; B	3; C	0; D	4; E	3; F	4; G
Node E	14; C	5; D	6; C	4; D	0; E	3; F	5; F
Node F	INF	4; D	6; D	3; D	3; E	0; F	2; G
Node G	22; B	5; D	7; D	4; D	5; F	2; F	0; G

In the next iteration,

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node A	0; A	12; B	8; C	11; C	14; C	14; C	15; C
Node B	12; A	0; B	4; D	1; D	5; D	4; D	5; D
Node C	8; A	4; D	0; C	3; D	6; E	6; D	7; D
Node D	11; C	1; B	3; C	0; D	4; E	3; F	4; G
Node E	14; C	5; D	6; C	4; D	0; E	3; F	5; F
Node F	14; D	4; D	6; D	3; D	3; E	0; F	2; G
Node G	15; D	5; D	7; D	4; D	5; F	2; F	0; G

If we compute the next iteration, there is no change in the matrix. Now, our table has converged. For Nodes B, D, and E, you should get the corresponding rows in the matrix for those nodes.

3. Once the link cost of D-C increases to 30, D and C first compute their own tables. Then, they pass the table to their neighbors for computing updated routing tables. With Distance Vector, bad news spreads slowly, so we will only show updates made to routing tables of B and D for brevity.

   Routing table updates for B:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node B	12; A	0; B	4; D	1; D	5; D	4; D	5; D

II:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node B	12; A	0; B	6; D	1; D	5; D	4; D	5; D

III:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node B	12; A	0; B	8; D	1; D	5; D	4; D	5; D

IV:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node B	12; A	0; B	9; C	1; D	5; D	4; D	5; D

B's routing table has converged.

Routing table updates for D:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node D	11; C	1; B	3; C	0; D	4; E	3; F	4; G

II:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node D	13; B	1; B	5; B	0; D	4; E	3; F	4; G

III:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node D	13; B	1; B	9; B	0; D	4; E	3; F	4; G

IV:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node D	13; B	1; B	9; B	0; D	4; E	3; F	4; G

V:

	Node A; Next hop	Node B; Next hop	Node C; Next hop	Node D; Next hop	Node E; Next hop	Node F; Next hop	Node G; Next hop
Node D	13; B	1; B	10; B	0; D	4; E	3; F	4; G

D’s routing table has converged.

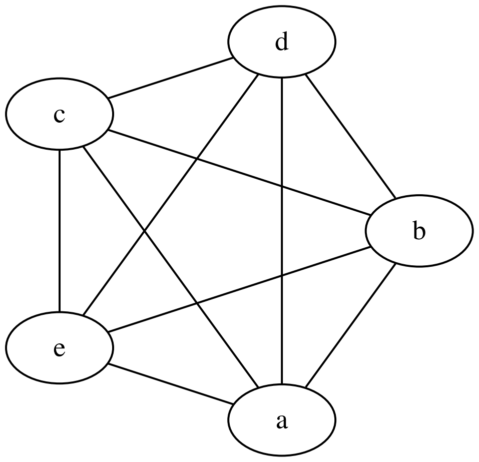
6 Routing Algorithms – 9 points

1. Consider the network of routers as shown in Fig. 3. Suppose you are told that traffic from a to b is routed by the path $a \rightarrow c \rightarrow e \rightarrow b$

   (a) List all the possible routes from d to e that could coexist with the above route from a to b on this network, if routing is performed using the link-state algorithm. Hint: Consider what the routing decision $a \rightarrow c \rightarrow e \rightarrow b$ says about the relative costs of individual edges. Can you use them to rule some items out?

   (b) List all the possible routes from d to a that could coexist with the above route from a to b on this network, if routing is performed using software- defined networking.

Figure 3:



2. Sort the following protocols by the amount of state each node maintains, and give clear explanation.

   - Link State
   - Distance Vector
   - Path Vector

Solution: (13 = 4+4+2+3 points)

1. (a) We know that link state basically computes least cost paths between nodes. In this graph we are not given any path costs. We are however given the path taken from a to b. Therefore the first hop from d could be any of the 4 other nodes (because we have no information regarding links between d and other nodes). There if d—e was a low cost path, then d—e would have been a potential path.

Suppose d–c was the first hop, then d—c—e would be the path. Since c—e is part of the path between a to b, this implies d—c—e will be the cheapest path. Following this logic, the possible paths in this routing algorithm are as provided below -

$$\begin{aligned} d \to a \to c \to e \\ d \to b \to e \\ d \to c \to e \\ d \to e \end{aligned}$$

(b) All (non looping) paths can exist because with SDN we can implement any routing condition we wish.

2. (a) $DV < PV < LS$.

DS only has to maintain information about its neighbours whereas LS has to store information about the costs of all endges in the network. Hence LS is the maximum whereas DS is the minimum. PV being somewhat of a mix of LS and DV, maintains less state than LS but more than DV.

DS is least because it has information about only its neighbours.

LS is maximum because it has link cost information about the entire network.

PV (used in BGP) is a mix of the above two. It has information of its own AS, but does not possess complete information of the whole network. Hence PV maintains state less than LS but more than DS (more than just the neighbours.)

Different order of PV and DV with appropriate reasoning is also acceptable.