EE673 Assignment 3

Adit Jain, 200038

October 20th, 2023

1 Solution 1

1.1 IP Lab

1. 192.168.1.46

2. ICMP (0x01)

3. There are 20 *bytes* in the IP header and the total length is 56 *bytes*. This makes the payload of the IP datagram to be 36 *bytes*.

4. More fragment bit is zero, hence data is not fragmented.

5. Identification, TTL and Header checksum always change

6. The fields that stay constant and must stay constant across the IP datagrams are:

   - Version (since IPv4 is being used for all packets)
   - Header length (since these are ICMP packets)
   - Source IP address (since we are sending from the same source)
   - Destination IP address (since we are sending to the same destination)
   - Differentiated Services (since all packets are ICMP, they use same Type of Service class)
   - Upper Layer Protocol (since these are ICMP packets)

   The fields that must change are:

   - Identification (IP packets must have different ids)
   - Time to live (traceroute increments each subsequent packets)
   - Header checksum (since header changes, so must checksum)

7. The pattern is that the IP header identification fields increment with each ICMP Echo (ping) request.

8. Identification: 30767
   TTL: 64

9. The identification field changes for all the ICMP TTL-exceeded replies because the identification field is a unique value. When two or more IP datagrams have the same identification value, then it means that these datagrams are fragments of a single large IP datagram.

   The TTL field remains the same or unchanged because the TTL for the first hop router is always the same.

10. Yes, this packet has been fragmented across more than one IP datagram.

11. The Flags bit for more fragments is set, indicating that the datagram has been fragmented. Since the fragment offset is 0, we know that this is the first fragment. This first datagram has a total length of 1500, including the header.

12. This is not the first fragment since the fragment offset is 1480. It is the last fragment since the more fragments flag is not set.

13. The IP header fields that changed between the fragments are:

    - Total length
    - Flags
    - Fragment offset
    - Checksum

14. After switching to 3500, there are 3 packets created from the original datagram.

15. The IP header fields that changed between all of the packets are:

    - Fragment
    - Offset
    - Checksum

    Between the first two packets and the last packet, we see a change in total length and also in the flags. The first two packets have a total length of 1500, with the more fragments bit set to 1, and the last packet has a total length of 540, with the more fragments bit set to 0.

1.2 ICMP Lab

1. Host: 192.168.1.101
   Destination Host: 143.89.14.34

2. The ICMP packet does not have source and destination port numbers because it was designed to communicate network layer information between hosts and routers, not between application layer processes. Each ICMP packet has a specific "Type" and a "Code". The Type/Code combination identifies the specific message being received. Since the network software itself interprets all ICMP messages, no port numbers are needed to direct the ICMP message to an application layer process.

3. The ICMP type is 8 and the code number is 0. The ICMP packet also has checksum, identifier, sequence number and data fields. The checksum, sequence number and identifier fields are two bytes each.

4. The ICMP type is 0 and the code number is 0. The ICMP packet also has checksum, identifier, sequence number and data fields. The checksum, sequence number and identifier fields are two bytes each.

5. Host: 192.168.1.101
   Destination Host: 138.96.146.2

6. No. If ICMP sent UDP packets instead, the IP protocol number should be 0x11.

7. The ICMP echo packet has the same fields as the ping query packets.

8. The ICMP error packet is not the same as the ping query packets. It contains both the IP header and the first 8 bytes of the original ICMP packet that the error is for.

9. The last three ICMP packets are message type 0 (echo reply) rather than 11 (TTL expired). They are different because the datagrams have made it all the way to the destination before the TTL expired.

10. There is a link between steps 11 and 12 that has a significantly longer delay. This is a transatlantic link from New York to Aubervillers, France. In figure 4 from the lab, the link is from New York to Pastourelle, France.

2 Solution 2

2.1 Link-State Routing Algorithm

The implementation for the same can be found in *solution2a.py*. The results I got from this are as shown below:

Destination	Cost	Next Hop
1	1	1
2	2	1
3	4	1

Node 0 Routing Table

Destination	Cost	Next Hop
0	1	0
2	1	2
3	3	2

Node 1 Routing Table

Destination	Cost	Next Hop
0	2	1
1	1	1
3	2	3

Node 2 Routing Table

Destination	Cost	Next Hop
0	4	2
1	3	2
2	2	2

Node 3 Routing Table

2.2 Distance-Vector Routing Algorithm

The implementation for the same can be found in *solution2b.py*. The results I got from this are as shown below:

	0	0	0	0
0	0	1	2	4
1	1	0	1	3
2	2	1	0	2
3	4	3	2	0

Min Cost

3 Solution 3

3.1 Markov Chain

In this 2 by 2 switch system, we have four possible states in the Markov chain, representing the combinations of head-of-line packets at input queues 1 and 2:

- **State 11:** Both the queues have packets destined for output port 1.
- **State 12:** Queue 1 has a packet destined for output port 1, and queue 2 has a packet destined for output port 2.
- **State 21:** Queue 1 has a packet destined for output port 2, and queue 2 has a packet destined for output port 1.
- **State 22:** Both the queues have packets destined for output port 2.

To construct the transition matrix, we need to define the transition probabilities between these states. Given that the destinations are chosen uniformly and independently, the transition probabilities are as follows:

- For State 11, there is a 50% probability that queue 1's packet will be moved and replaced by a new packet destined for port 1. Similarly, it has a 50% probability of receiving a new packet requesting port 2. For queue 2, the same probability apply. This gives us the following transition probabilities:
  - $P(11 \to 11) = 1/2$
  - $P(11 \to 12) = 1/4$
  - $P(11 \to 21) = 1/4$
- For State 12, both packets are transferable. The new locations are chosen independently and consistently:
  - $P(12 \to 11) = 1/4$
  - $P(12 \to 12) = 1/4$
  - $P(12 \to 21) = 1/4$
  - $P(12 \to 22) = 1/4$
- For State 21, the transition probabilities will be symmetric to state 12:
  - $P(21 \to 11) = 1/4$

- $P(21 \to 12) = 1/4$
- $P(21 \to 21) = 1/4$
- $P(21 \to 22) = 1/4$

- For State 22, the transition probabilities
  - $P(22 \to 12) = 1/4$
  - $P(22 \to 21) = 1/4$
  - $P(22 \to 22) = 1/2$

The transition matrix for this Markov chain is as follows:

	State 11	State 12	State 21	State 22
State 11	0.5	0.25	0.25	0
State 12	0.25	0.25	0.25	0.25
State 21	0.25	0.25	0.25	0.25
State 22	0	0.25	0.25	0.5

To compute the stationary distribution of the Markov chain, we need to find the eigenvector corresponding to the eigenvalue of 1 for the transition matrix (P). For that, we need to find n_{1x4} such that it satisfies the following condition:

$$nP = n \tag{1}$$

Using the code as provided in *solution3.py*, n was found out to be $[0.25, 0.25, 0.25, 0.25]^T$ post normalization, which represents the stationary distribution of the Markov chain. This also displays the Independence and uniformity of the events.

3.2 Average Throughput

Based on the above calculated stationary distribution, we can easily calculate the average throughput of the provided 2x2 switch, which is given by:

$$\begin{aligned} T &= P_{11} + P_{22} + 0.5(P_{12} + P_{21}) \\ &= 0.25 + 0.25 + 0.5(0.25 + 0.25) \\ &= 0.75 \end{aligned} \tag{2}$$

Hence, average throughput = 0.75.

4 Solution 4

4.1 Capacity Region

To draw the capacity region of this wireless network, we need to consider the achievable rated for both mobiles in each of the two channel states. Given the channel states:

$$\begin{aligned} c_1 &= (1,2) \\ c_2 &= (3,1) \end{aligned} \tag{3}$$

Let's denote the rates for mobile 1 and mobile 2 by R_1 and R_2 respectively. Now, in channel state c_1,

- If mobile 1 is selected, it gets a rate of $c(1) = 1\ packets/sec$
- If mobile 2 is selected, it gets a rate of $c(2) = 2\ packets/sec$

In channel state c_2,

- If mobile 1 is selected, it gets a rate of $c(1) = 3\ packets/sec$
- If mobile 2 is selected, it gets a rate of $c(2) = 1\ packets/sec$

The capacity region will be the set of achievable rates (R_1, R_2) that satisfy the following conditions:

- $R_1 \leq 1$ (from c_1)
- $R_2 \leq 2$ (from c_1)
- $R_1 \leq 3$ (from c_2)
- $R_2 \leq 1$ (from c_2)

Additionally, both R_1 and R_2 should be non-negative. The capacitive region can be represented as a polygonal line in the (R_1, R_2) plane. To draw it, we need to plot the points (1,2), (3,1), (1,0) and (0,0) and connect them to form a polygon. The corners of the polygon represent the achievable rate pairs (R_1, R_2) for different combinations of the channel states and mobile selections. Code that is provided in *solution4a.py* was used to plot fig 1.

4.2 Stability Analysis

Given the provided system, let's denote the Lyapunov function as:

$$V(t) = q_1(t)^2 + q_2(t)^2 \tag{4}$$

where $q = [q_1, q_2]$ is the vector of queue lengths. On analysing the Lyapunov drift, we get:

$$\begin{aligned} &\mathbb{E}[V(t{+}1) - V(t)|q(t)] \\ &\quad= \mathbb{E}[(q_1(t+1)^2 + q_2(t+1)^2) - (q_1(t)^2 + q_2(t)^2)|q(t)] \\ &\quad= \mathbb{E}[(q_1(t) + a_1(t) - d_1(t))^2 + (q_2(t) + a_2(t) - d_2(t))^2 - (q_1(t)^2 + q_2(t)^2)|q(t)] \\ &\quad= \mathbb{E}[2(q_1(t) + a_1(t) - d_1(t))(a_1(t) - d_1(t)) + 2(q_2(t) + a_2(t) - d_2(t))(a_2(t) - d_2(t)) + \\ &\quad(a_1(t) - d_1(t))^2 + (a_2(t) - d_2(t))^2|q(t)] \\ &\quad= 2\mathbb{E}[(q_1(t) + a_1(t) - d_1(t))(a_1(t) - d_1(t)) + (q_2(t) + a_2(t) - d_2(t))(a_2(t) - d_2(t))] + \\ &\quad\mathbb{E}[(a_1(t) - d_1(t))^2 + (a_2(t) - d_2(t))^2] \end{aligned} \tag{5}$$

Analysing the term $(q_i(t) + a_i(t) - d_i(t))(a_i(t) - d_i(t))$, we can break it down into two cases:

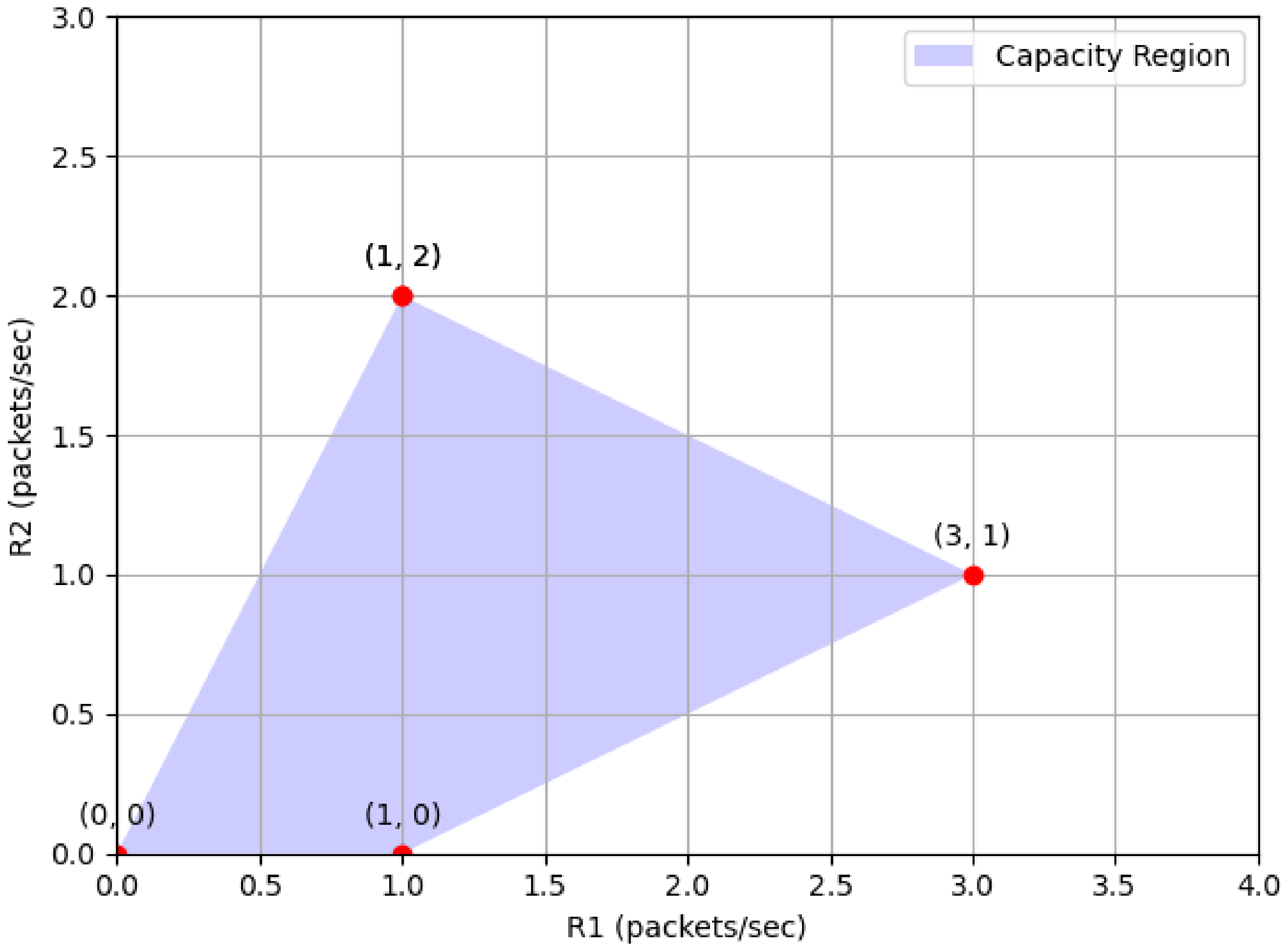

Figure 1: Capacity region of the wireless network

1. If $a_i(t) \leq d_i(t)$, then $(q_i(t) + a_i(t) - d_i(t))(a_i(t) - d_i(t)) \leq 0$
2. If $a_i(t) \geq d_i(t)$, then $(q_i(t) + a_i(t) - d_i(t))(a_i(t) - d_i(t)) = (q_i(t) + a_i(t) - d_i(t))(a_i(t) - d_i(t))$

This implies that the Lyapunov drift is less than or equal to:

$2\mathbb{E}[(q_1(t)+a_1(t)-d_1(t))(a_1(t)-d_1(t))+(q_2(t)+a_2(t)-d_2(t))(a_2(t)-d_2(t))]+\mathbb{E}[(a_1(t)-d_1(t))^2+(a_2(t)-d_2(t))^2]$

Now, since we are using the maximum-weight scheduling algorithm, user $j(t)$ is selected in each time slot, which means $q_j(t) + a_j(t) - d_j(t) \geq q_i(t) + a_i(t) - d_i(t)$ for all i.
As a result, we can bound the Lyapunov drift as:

$$\mathbb{E}[V(t+1) - V(t)|q(t)] \leq 2\mathbb{E}[(q_j(t) + a_j(t) - d_j(t))(a_j(t) - d_j(t))] + \mathbb{E}[(a_j(t) - d_j(t))^2] \qquad (6)$$

This shows that the Lyapunov drift is bounded, and by the Foster-Lyapunov theorum, the system is stable using the Lyapunov function $V(t) = q_1(t)^2 + q_2(t)^2$.