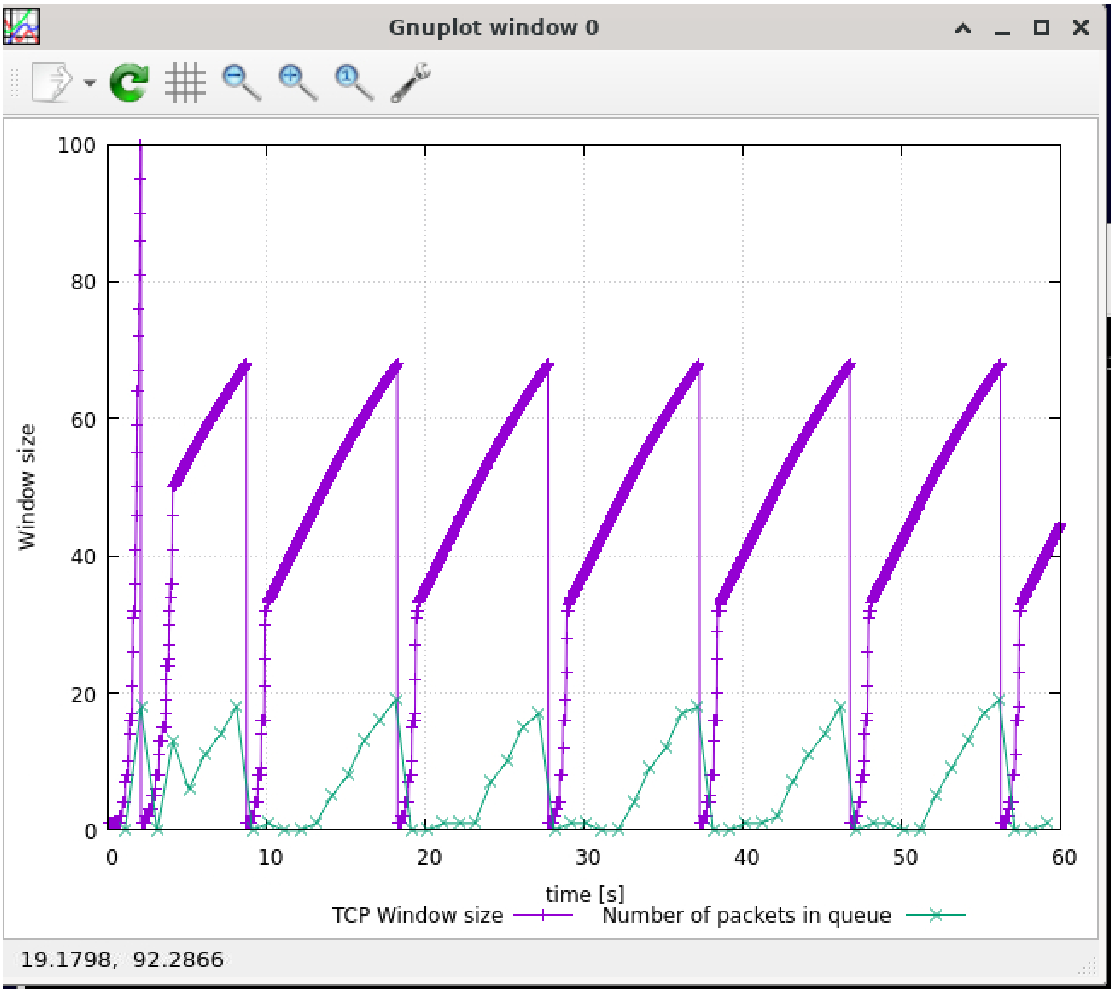
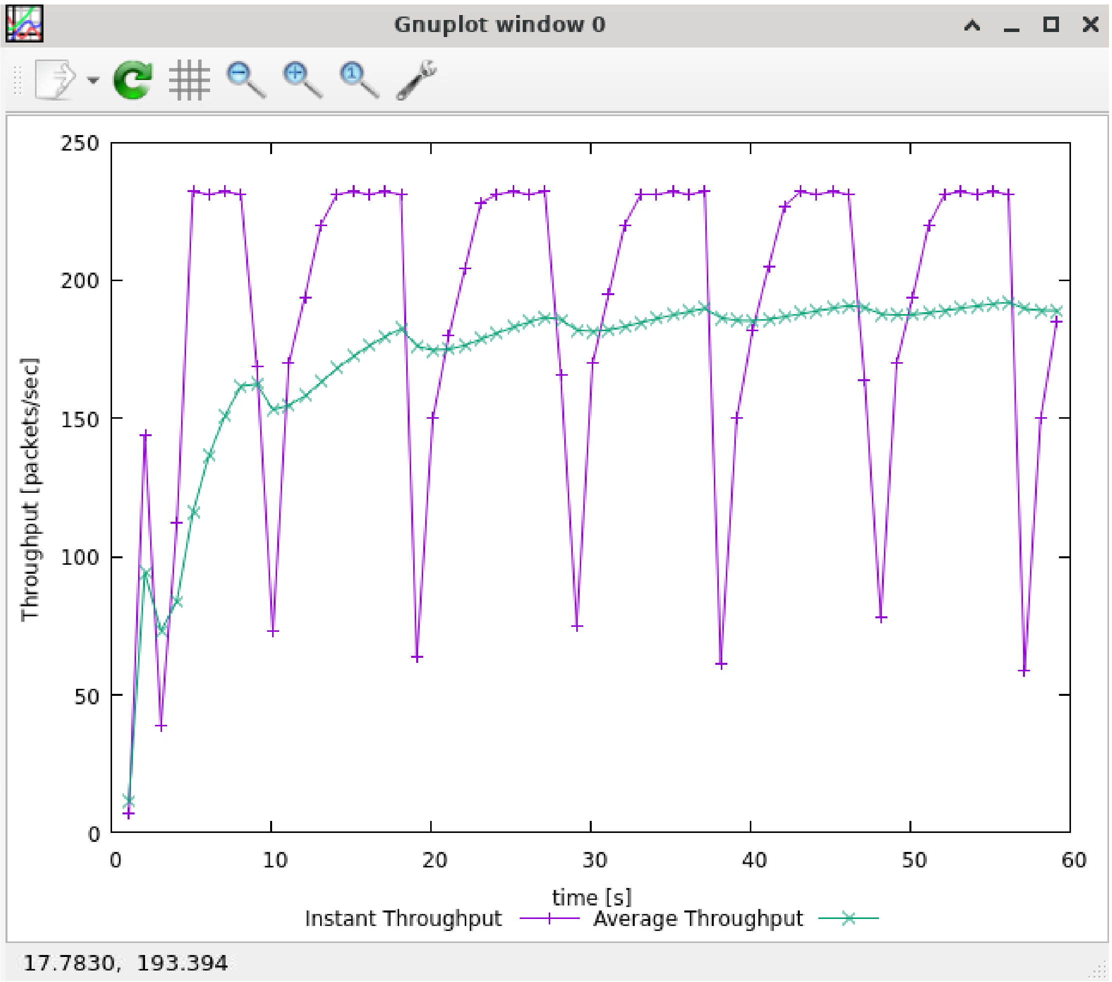
Exercise 1



Q1:

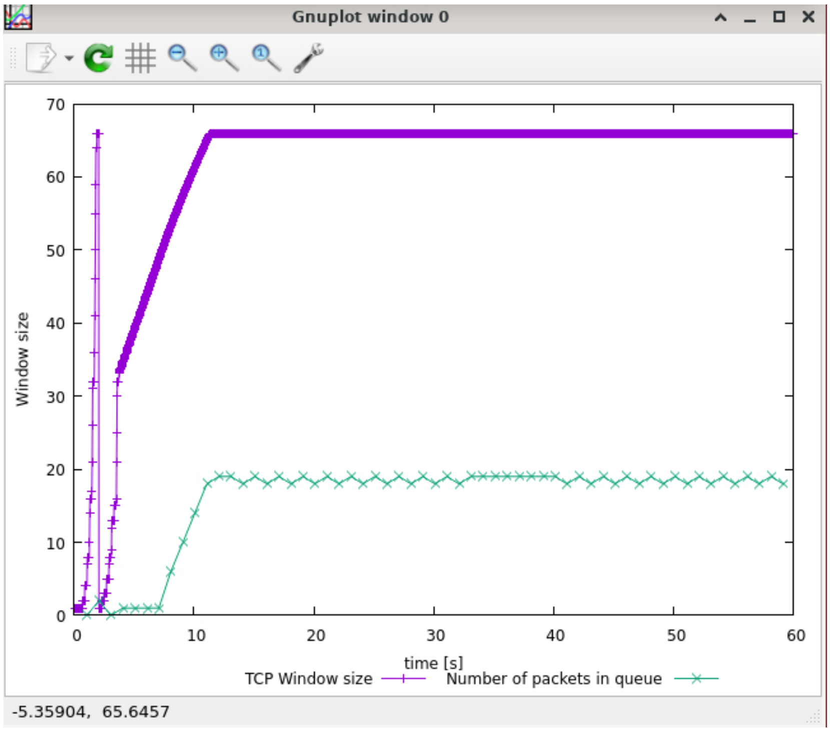
- The maximum congestion window size reached by a TCP flow is 100.
- Although the maximum window is set to 150, the queueSize in tpWindow.tcl is 20, so when the queue is full, the rest of the packets are discarded.
- This causes the sender to reduce the congestion window to 1. The threshold is set to one-half of the current window size, which is 50. It then re-enters the slow start, waits until it reaches 50 and then enters the AIMD phase, and so on.

Q2:

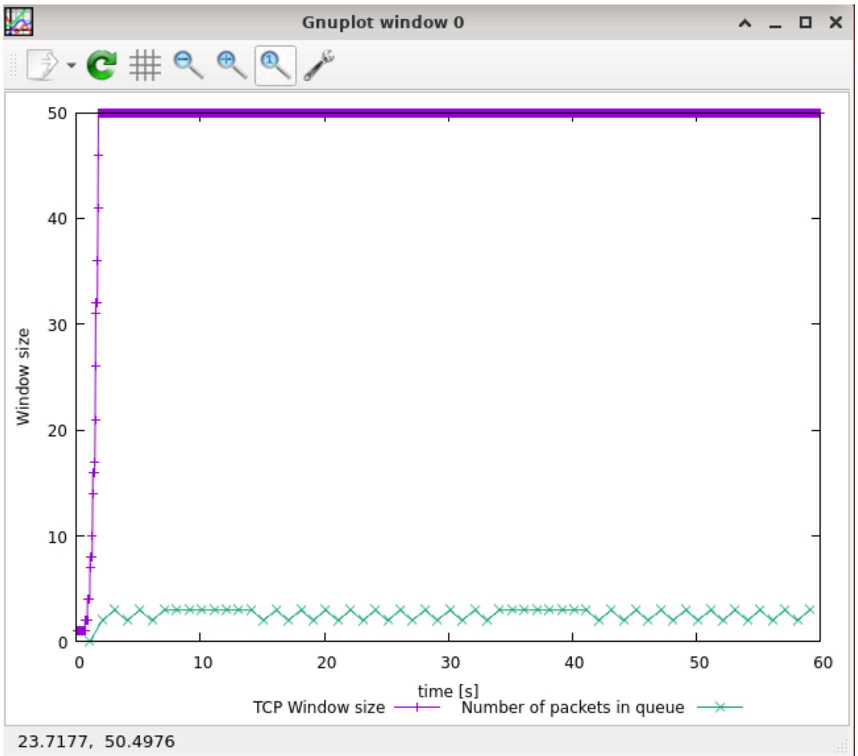


- The average throughput of TCP in packets per second is 190.
- The average throughput of TCP in bit per second without any headers is:
  - 190 * 500 * 8 = 760,000
- The average throughput of TCP in bit per second with IP and TCP headers is:
  - 190 * (500 + 40) * 8 = 820,800

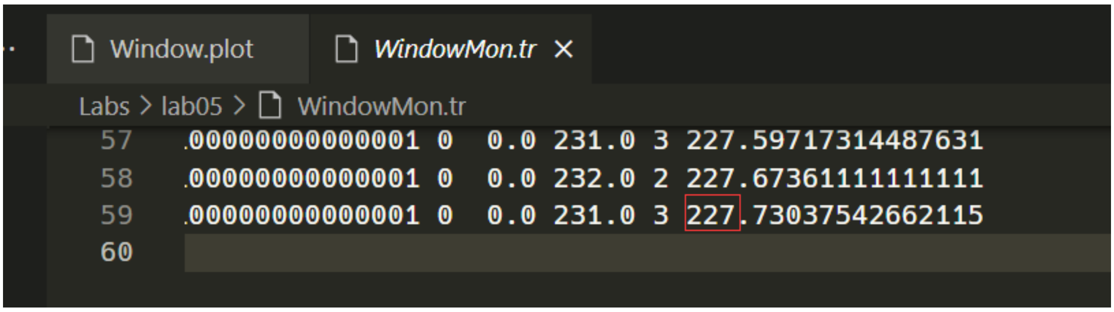
Q3:



- When the max congestion window size is set to <=66, the queue is no longer full, so packets are no longer loss.



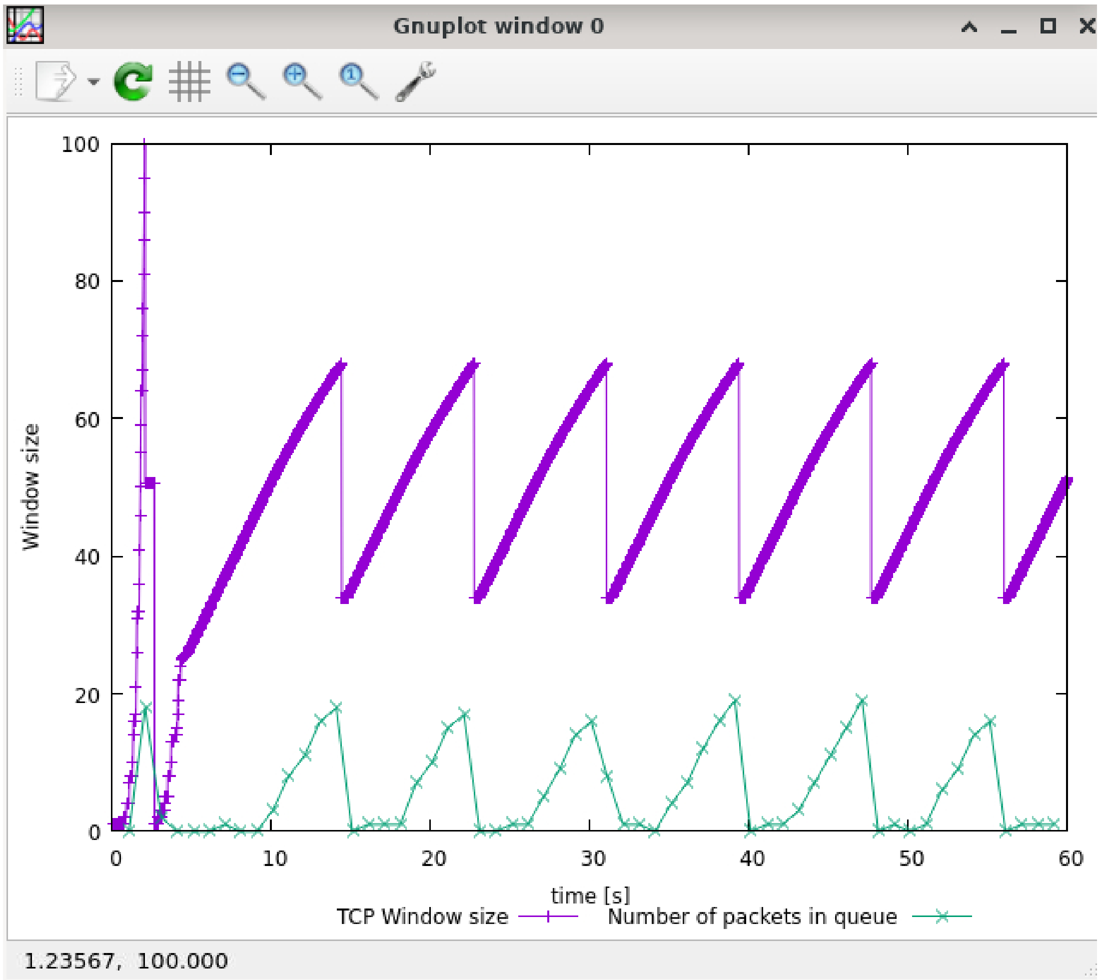
- When the max congestion window size is set to <=50, TCP stops oscillating.

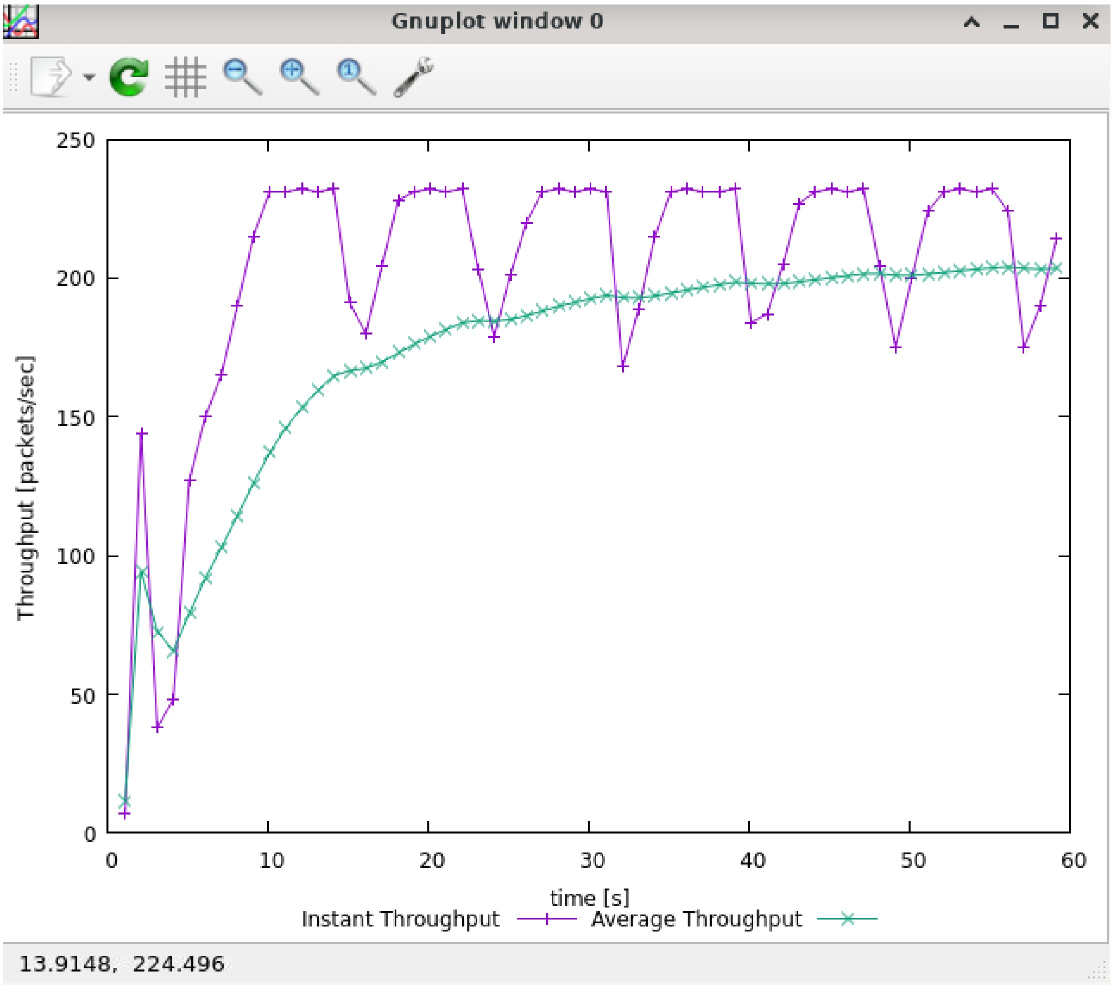


```
Window.plot   WindowMon.tr ×
Labs > lab05 > WindowMon.tr
57   .000000000000001 0  0.0 231.0 3 227.59717314487631
58   .000000000000001 0  0.0 232.0 2 227.67361111111111
59   .000000000000001 0  0.0 231.0 3 227.73037542662115
60
```

- The average throughput in packets per second at this point is 227, and in bit per second is 227 * 500 * 8 = 908,000.
- This throughput is very close to the link capacity (1Mbps).

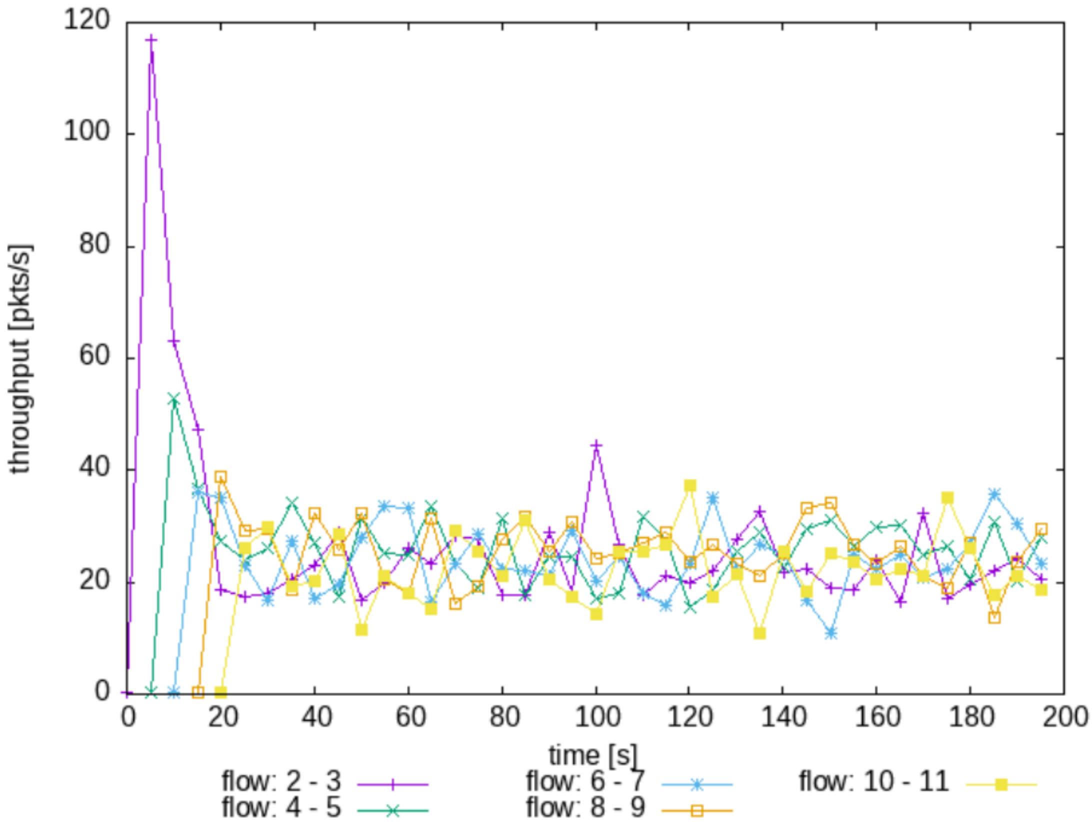
Q4:





- Comparing the diagrams of the two implementations, we can see that, except for a timeout event at the beginning that causes the current window size to be set to 1. After that, TCP does not enter the slow start phase anymore. Only the AIMD phase is repeated over and over again. This is because in TCP Reno, if the sender receives three duplicate ACKs, it will only reduce the window size to 1/2 of the current size.
- The average throughput of TCP Reno is 203, which is higher than TCP Tahoe's 190 because TCP Reno omits the slow start phase for three duplicate ACKs.

Exercise 2



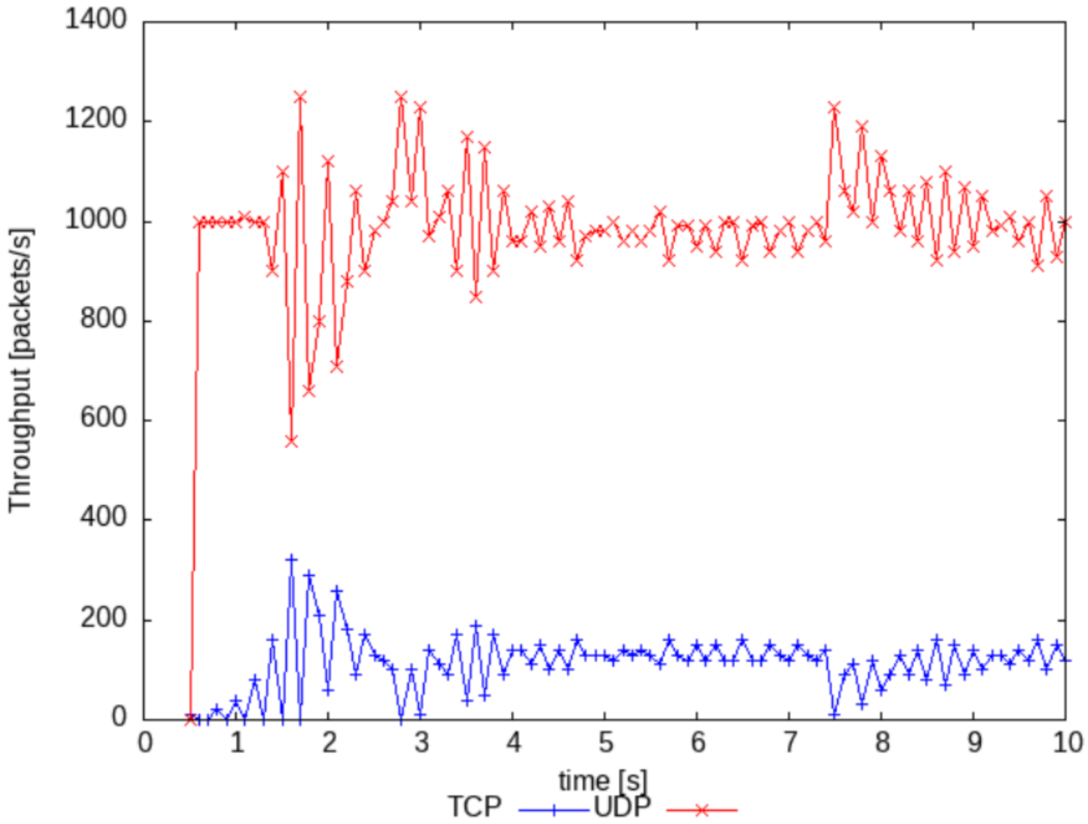
Q1:

- Each flow gets an equal share of the capacity of the common link.
- We can see from the graph that all connections start at 20 seconds.
- The throughput of all five connections is similar according to the AIMD control algorithm.

Q2:

- Whenever a new flow is created, it causes the window size of the previous flow to decrease because congestion occurs.
- It is fair that whenever a new flow is joined, all existing flows should have a fair share of the total traffic.

Exercise 3



Q1:

- UDP should not reduce its own transmission rate due to congestion.
- TCP will reduce its own transmission rate due to congestion control and will not compete with UDP.

Q2:

- It is normal for UDP flows to have higher throughput than TCP, because UDP is completely unconstrained by congestion control. This means that UDP sends packets at the same rate regardless of whether they are lost or not.
- TCP flows, on the other hand, will reduce their throughput once packet

loss is detected, so they must not be able to compete with UDP flows.

Q3:

- The advantage of using UDP flows when competing with others is that you can transmit data without restriction regardless of congestion. The disadvantage is that packets may be lost, so additional efforts are needed to implement reliable transmission for UDP.
- If everyone starts using UDP, the end result is that the network crashes and the transmission speed for everyone becomes extremely low. Because once there is packet loss everyone retransmits more packets, and then the router queue fills up and enters a vicious cycle. This sort of thing has happened throughout history.