Computer Networks

Q.1.

a. Maximum expected (theoretical) value of throughput is 5Mbps which is the bottleneck of the two (between N0, N1 and between N1,N2) 10Mbps and 5Mbps respectively.

b. Bandwidth-Delay-Product is the product of Bandwidth and Round Trip time

BDP = Bandwidth * RTT

RTT = (delay1 + delay2) * 2 = (10ms + 15ms) *2 = 50ms

Therefore,

BDP = 5 Mbps(Bandwidth) * 50ms(RTT)

= 250 * 1000 * 1000 * 1/1000 bits

= 250000 bits

In packets = Bits / Application payload size

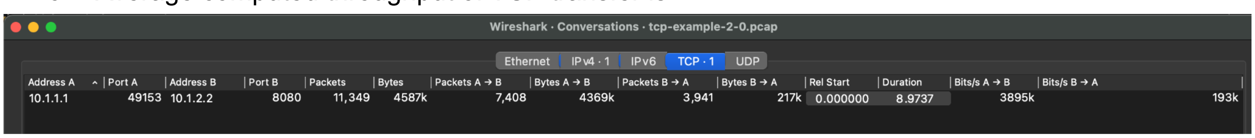
= 250000 / (1460 Bytes)

= 250000 / (1460 * 8)

= 21.404 packets

Since we cannot transmit 21.404 packets, the number of packets transmitted would be 22.

c. Average computed throughput of TCP transfer is


Wireshark · Conversations · tcp-example-2-0.pcap

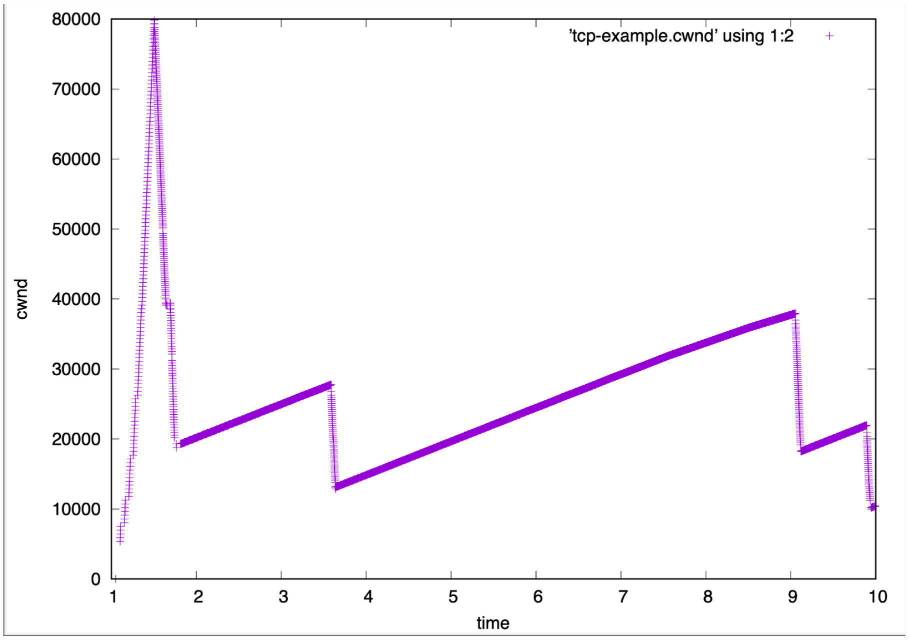
Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,349	4587k	7,408	4369k	3,941	217k	0.000000	8.9737	3895k	193k

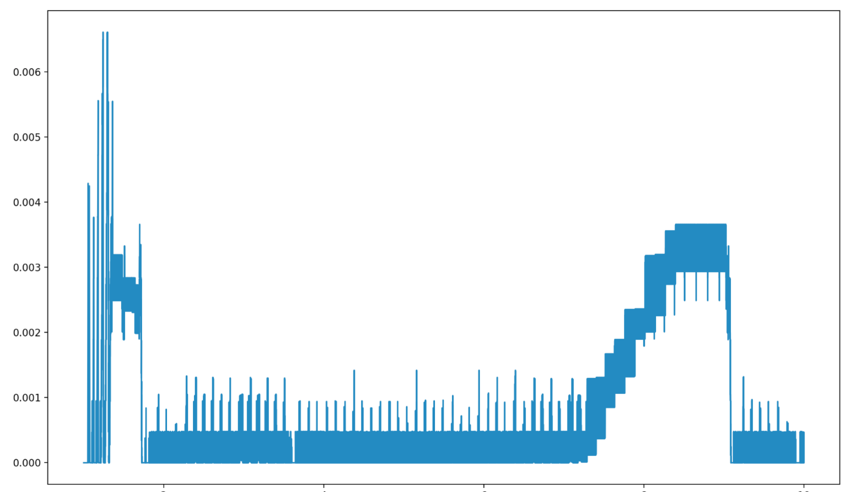
Average = 3895Kbps + 193Kbps = 4088Kbps

d. Maximum theoretical throughput value is greater than the value observed because there are other factors than just bottleneck bandwidth which affect the throughput. These could be packet errors and queue size. Packets can be dropped due to errors in transmission or due to a full queue. This would need retransmission of the packets which would cause some delays. The packets once in the queue may also face queueing delays which would further reduce the throughput.

e. Plot is as follows



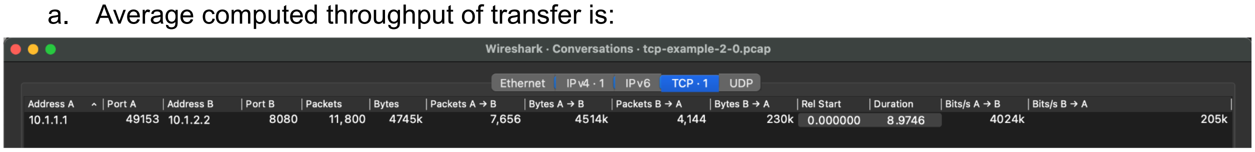
f. Queueing delay with respect to time is



g. Plots 1(e) and 1(f) have very similar trajectories. They both rise initially and then fall after that it slowly rises. This is because congestion is directly related to the queueing delay. A larger queueing delay would mean there is more congestion.

Q.2.

a. Average computed throughput of transfer is:



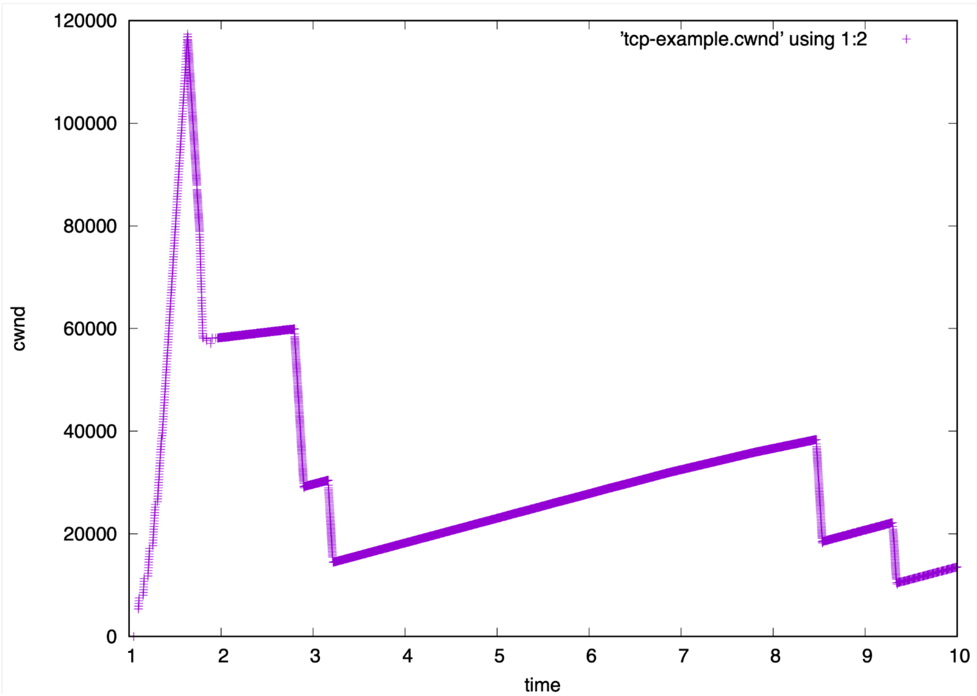
Wireshark · Conversations · tcp-example-2-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

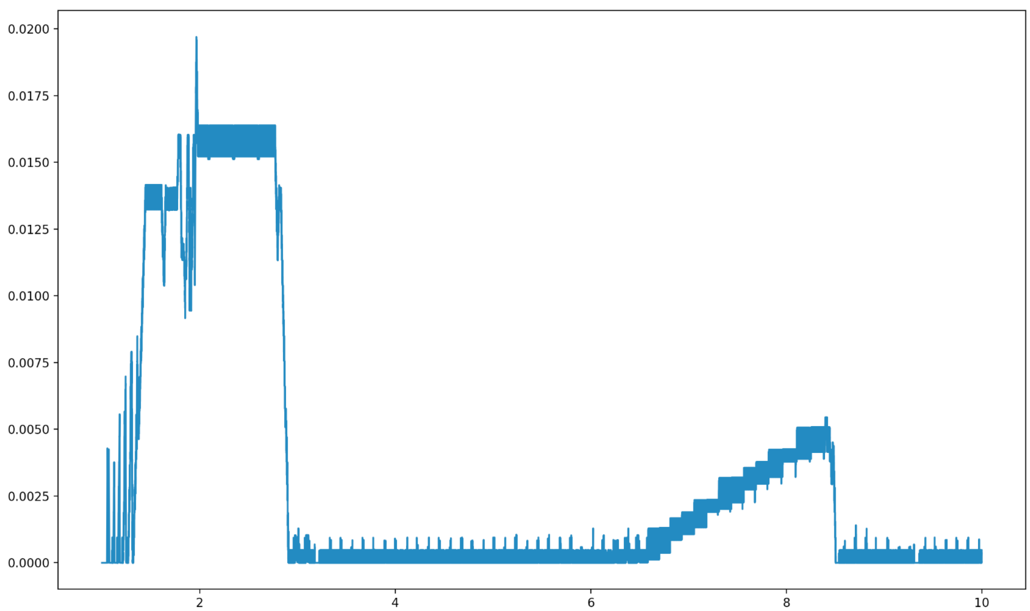
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	11,800	4745k	7,656	4514k	4,144	230k	0.000000	8.9746	4024k	205k

Average = 4024Kbps + 205Kbps = 4229Kbps

b. CWND with time



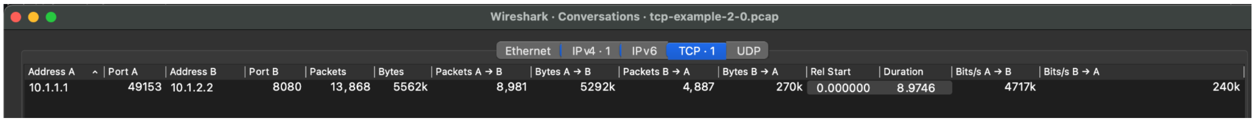
c. Queueing delay with time



d. The maximum CWND value for the second one is 120000 which is more than the first question (80000). This is because with a larger queue size, we can queue more packets. Due to this the congestion window size increases and therefore we see a larger spike in this question.

Q.3.

a. Average computed throughput:



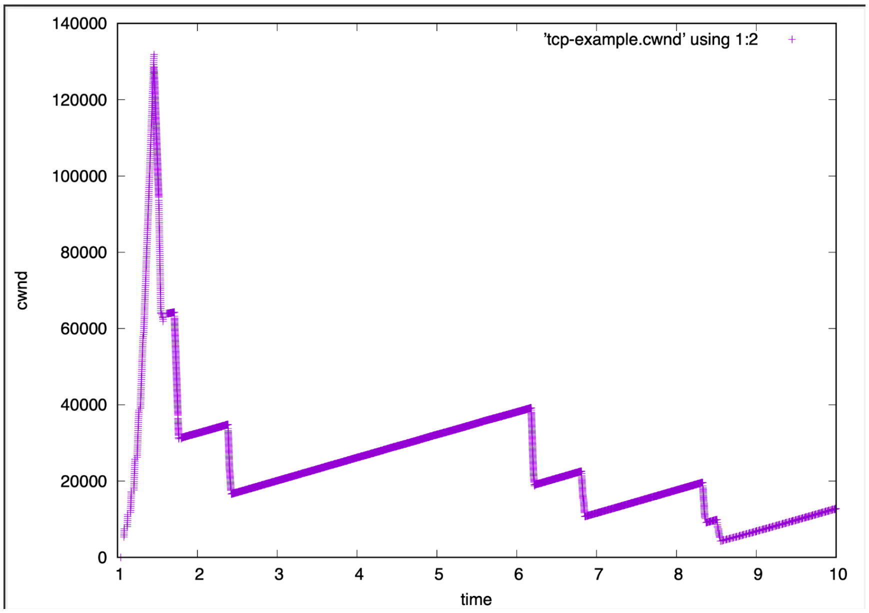
Wireshark · Conversations · tcp-example-2-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

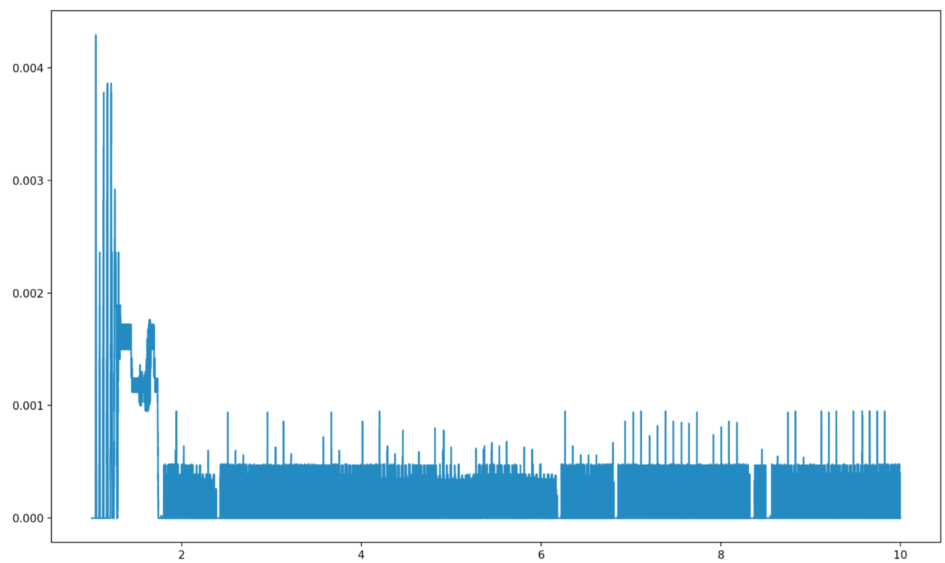
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	13,868	5562k	8,981	5292k	4,887	270k	0.000000	8.9746	4717k	240k

Average = 4717Kbps + 240Kbps = 4957Kbps

b. CWND with time



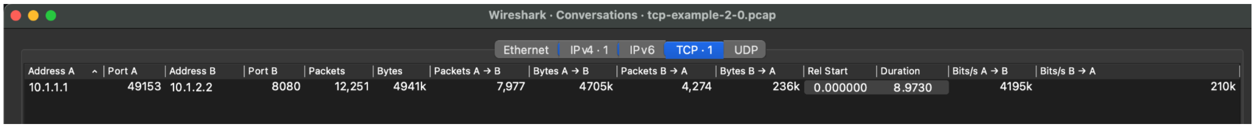
c. Queueing delay with time



d. The queuing delay in Q3 is less than that of Q1. This is because with an increased bandwidth, the number of packets waiting in the queue decreases, hence leading to lesser queue delays.

Q.4

a. Average computed throughput of TCP transfer

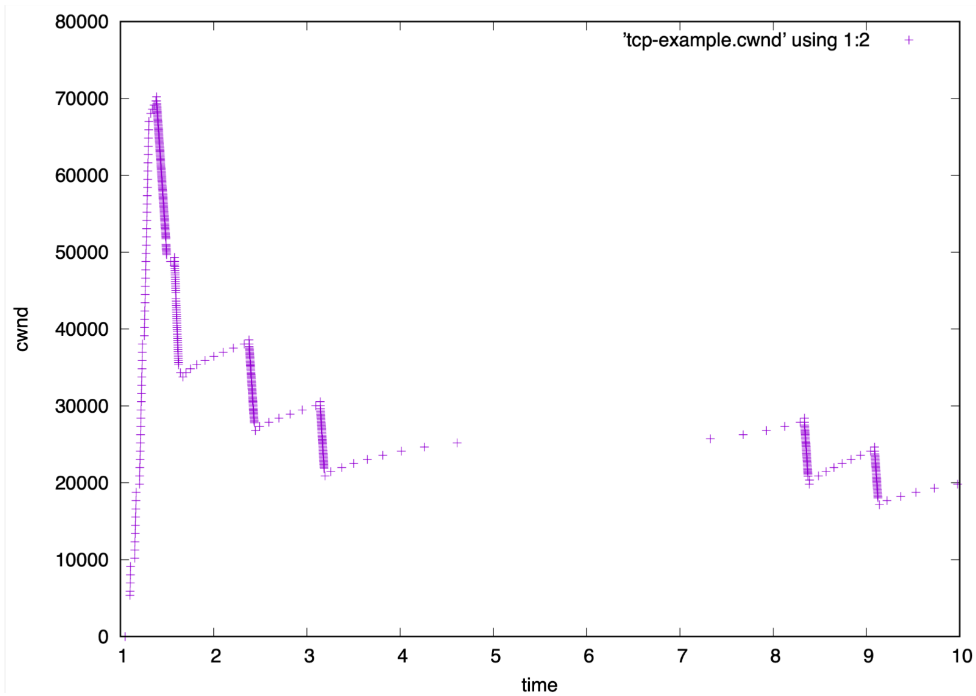

Wireshark · Conversations · tcp-example-2-0.pcap

Ethernet | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.1.1.1	49153	10.1.2.2	8080	12,251	4941k	7,977	4705k	4,274	236k	0.000000	8.9730	4195k	210k

Average = 4195Kbps + 210Kbps = 4405Kbps

b. CWND with time



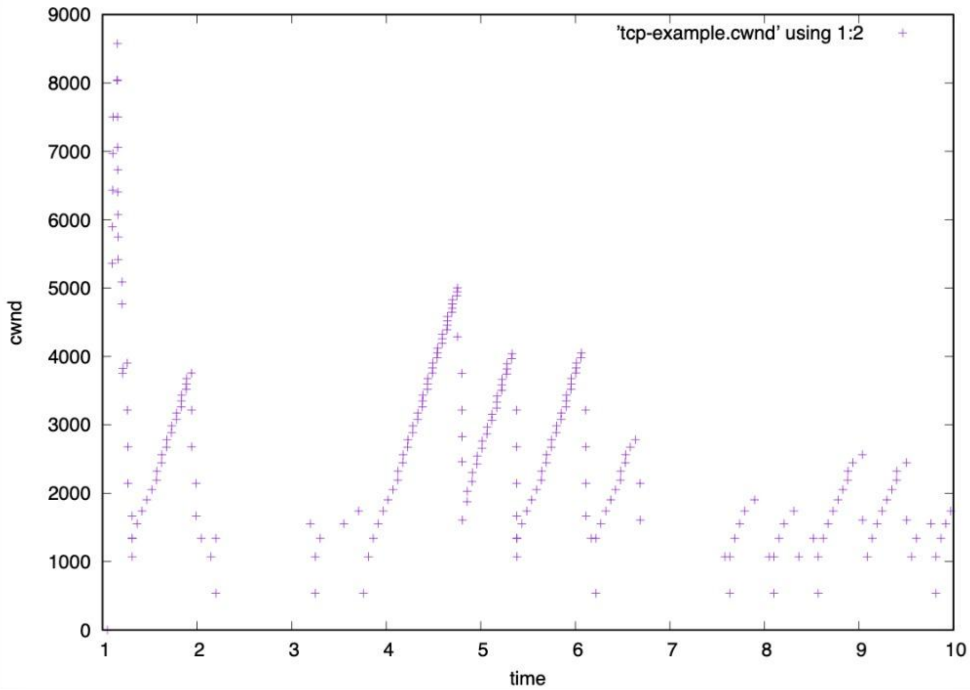
c. The graph obtained in Q1 is by using the TCP variant TcpNewReno, whereas in this question the variant used is TcpCubic. The difference between the two in various stages is as follows:

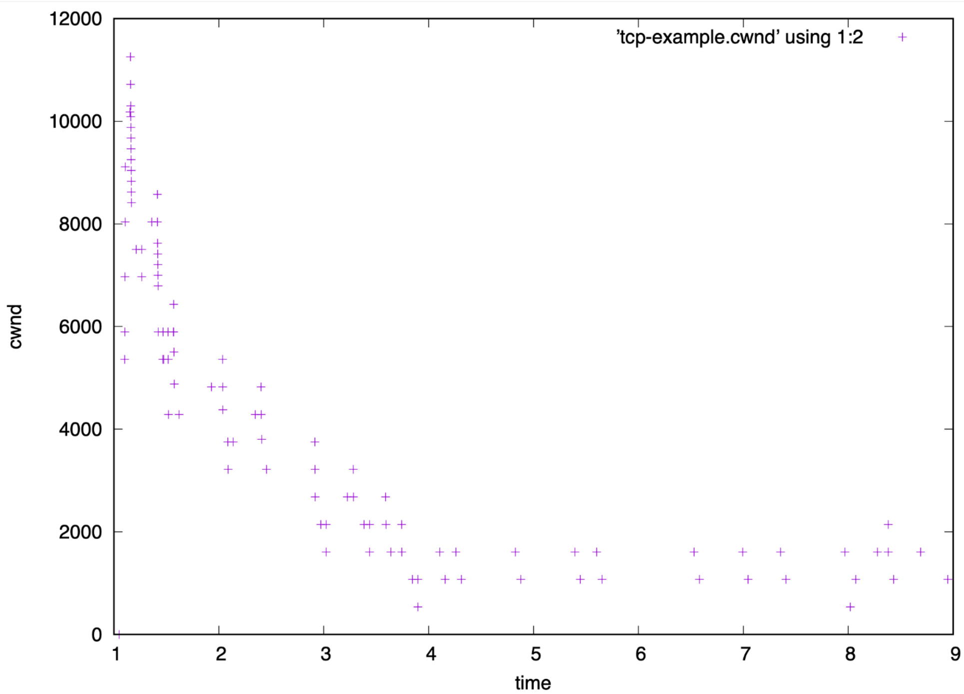
i. **Slow Start**: The graph in Q1 reaches a higher value which is 80000, compared to the value reached in this question which is 70000. The TcpCubic value however shows a much sharper increase than compared to TcpNewReno. This shows us that TcpNewReno has a slower start than TcpCubic.

ii. **Congestion Avoidance**: By comparing the two graphs in terms of congestion avoidance, we see that TcpCubic performs better. The graph is linear in TcpNewReno after the threshold value, whereas there is a cubic increase after the threshold in TcpCubic.

iii. **Fast Recovery**:TcpCubic has a faster recovery than TcpNewReno, from the graph we see that TcpCubic falls more often than TcpNewReno. TcpCubic has more fast recovery sections than TcpNewReno.

By changing the error rate to 0.0001. I got the following two graphs for TcpNewReno and TcpCubic respectively.





These graphs shown above show more clearly how the factors are affected by the two algorithms.