Assignment 1
Shreyansh Singh
2020CS10385

Description of files:

1) task1.cc : This file contains the code for Task1. This file is modified version of fifth.cc given in the tutorial. I have modified the parameters (like application data rate, channel data rate, packet size, etc. as per need) and added the command to make a cwnd file, using which congestion.plt drwas the graph. To count the number of packet drops I have added a counter in the function RxDrop.

2) task2.cc : This file contains the code for Task2. It is similar to task1 file just the application and channel data rate is changed.

3) task3.cc : This file contains the code for Task3. It is modified version of socket-bound-tcp-static-routing given in examples folder, instead of static routing, I have used dynamic routing. I have used MyApp class given in fifth.cc. To change UDP application rate, I have added a condition in ScheduleTx function to change the data rate when time reaches 30 seconds.

4) congestion.plt : This file plots the congestion window size vs time plot.

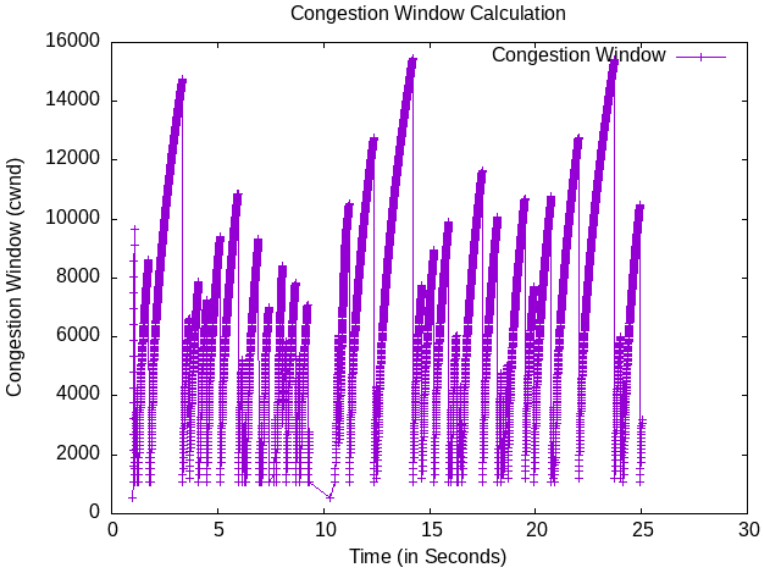
5) pcap files: pcap files generated for Task3.

Task 1:

1)

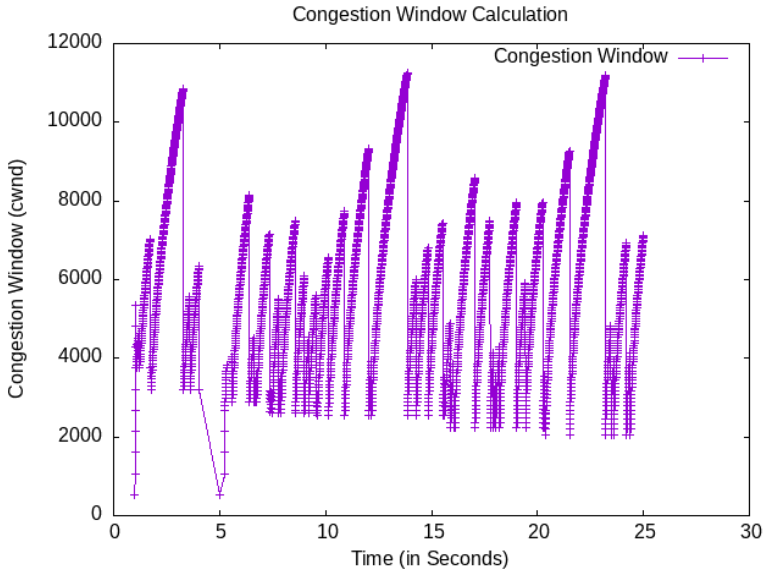
Protocal Name	Max. Window Size	No. of Packets drop
New Reno	15462	58
Vegas	11252	58
Veno	15462	59
WestWood	15471	60

2)



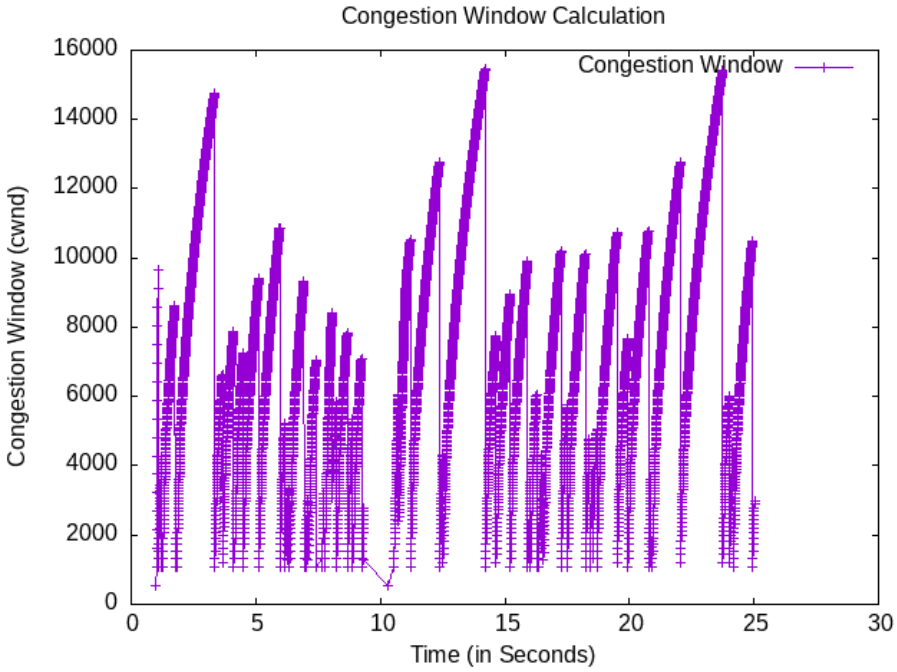
TCP NewReno

TCP NewReno is an update over Reno congestion control. Reno halves the size of congestion window for each packet loss, if there are many drops in a congestion window then thesize will be halved for each drop, this leads to a sudden drop in window size. To overcome this NewReno uses partial acknowledgement. As a result for every drop the congestion window size is not halved. In case of drop in a congestion window the size of congestion window is set to 1072. When multiple packets are dropped the size is further reduced to 536.



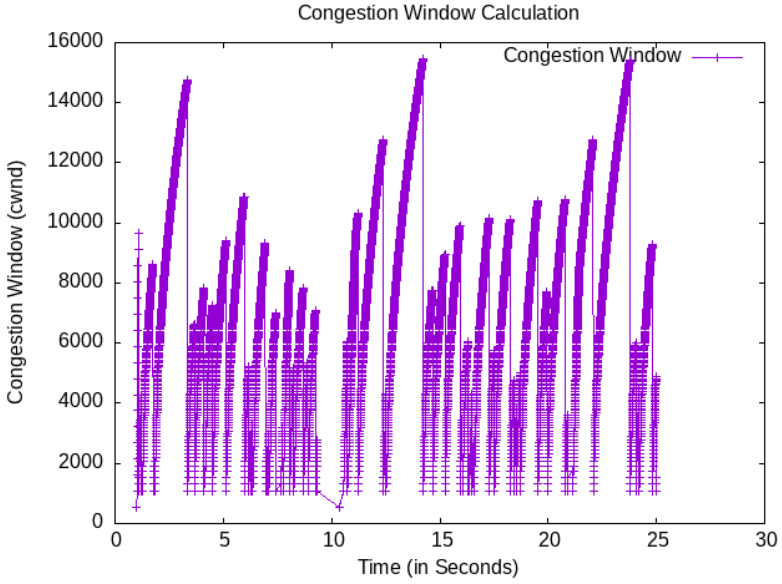
TCP Vegas

TCP Vegas detects congestion based on RTT (Round Trip Time) instead of actual packet loss. Thus it emphasizes on packet delay rather than packet loss. This is done to get idea about the rate at which to send packets.
The maximum congestion window size is less than other algorithms since it does not wait for actual packet loss.



TCP Veno

TCP Veno enhances Reno algorithm for dealing with random packet loss in wireless access networks by employing Vegas's method in estimating the backlog at the bottleneck queue to distinguish between congestive and non-congestive states. Veno decide the congestion window size on the calculated N and its predefined threshold beta. Where, N = Actual * (RTT – BaseRTT).
The graph is similar to that of NewReno and in case of packet drop it reduces the congestion window size to 1072 in most cases.



TCP WestWood

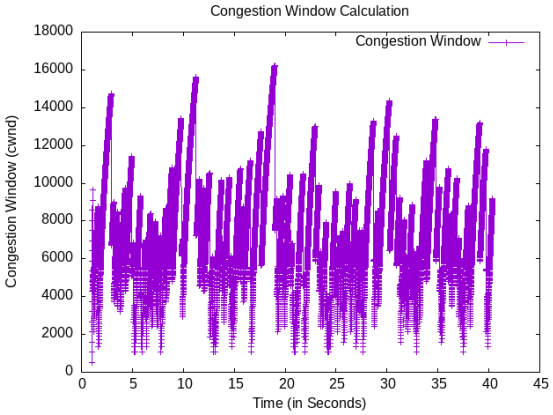
Westwood uses AIAD (Additive Increase/Adaptive Decrease) congestion control mechanism. In case of congestion it estimates the network's bandwidth and uses it to decide the size of congestion window.
The graph is similar to that of NewReno and in case of packet drop it reduces the congestion window size to 1072 in most cases.

3) The graphs of TCP NewReno, Veno, Westwood are very similar to one another whereas Vegas protocol is different from rest of the algorithms beacuse it is based on RTT (Round Trip Time) instead of actual packet loss. Thus the maximum congestion window size in case of Vegas (~11000) is much less than that of others (~15500) because Vegas decreases window size before actual packet drop when it senses increased RTT. There is slight difference in the number of packets drop and maximum window size of Westwood is slight different from other two and the rate of increase of window size in Westwood is slightly less than that of NewReno and Veno.

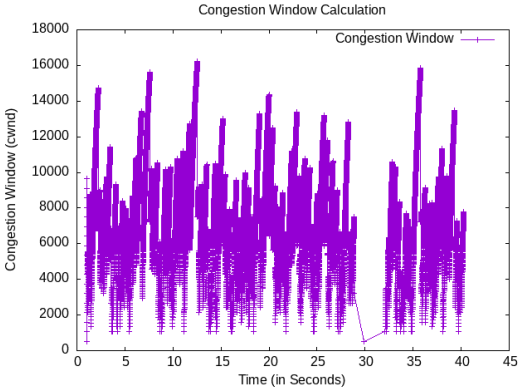
4) BBR ("Bottleneck Bandwidth and Round-trip propagation time") is a new congestion control algorithm developed at Google. All the previous algorithms used loss based congestion control, the size of congestion window is reduced with each packet loss eventhough their is no actual congestion at the bottleneck. BBR uses latency, instead of lost packets as a primary factor to determine the sending rate. BBR does not attempt to fill the buffers thus it tends to be better in avoiding buffer bloat.

Task 2:

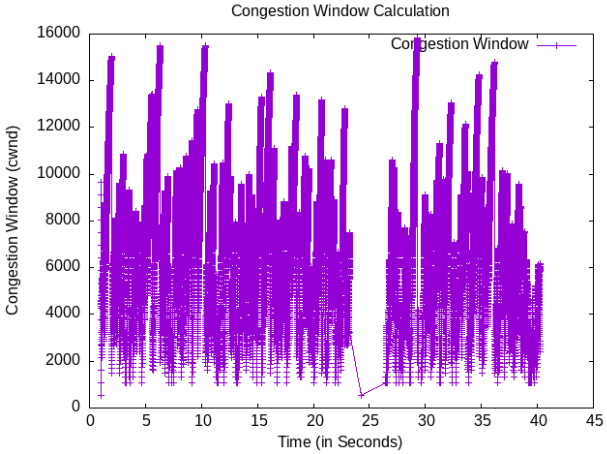
a)



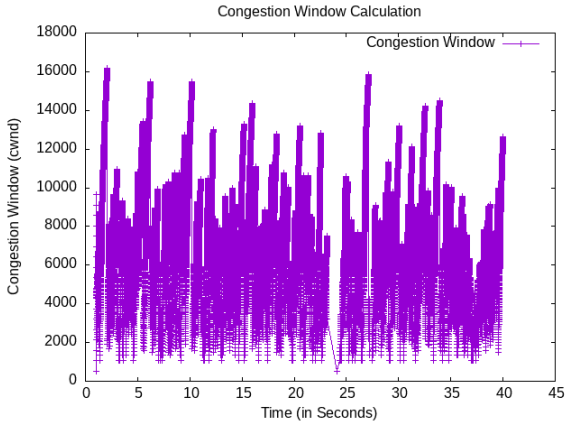
Channel Data Rate: 3Mbps



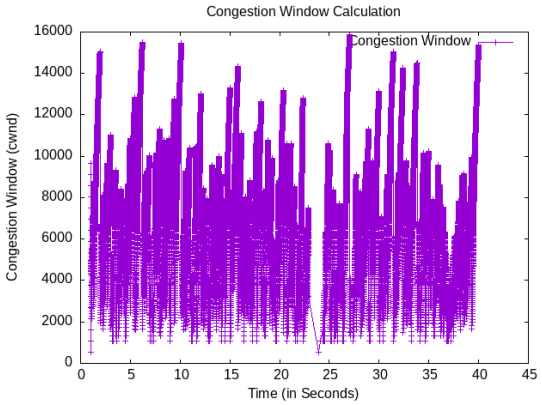
Channel Data Rate: 5Mbps



Channel Data Rate: 10Mbps

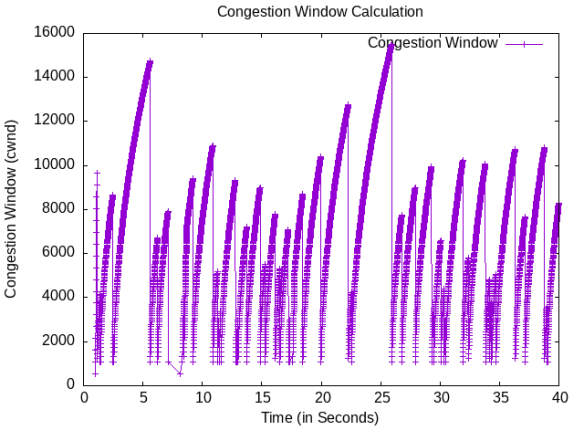


Channel Data Rate: 15Mbps

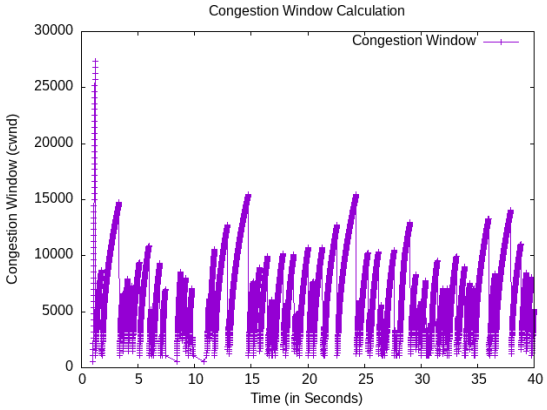


Channel Data Rate: 30Mbps

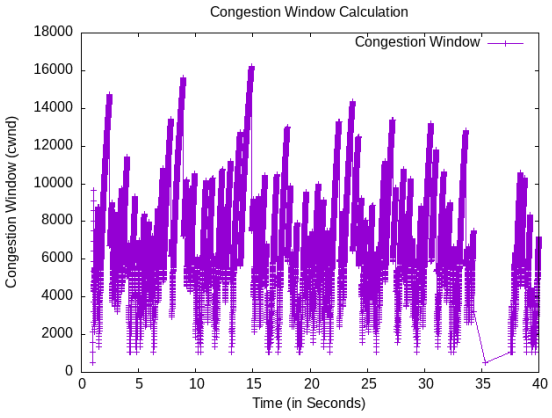
b)



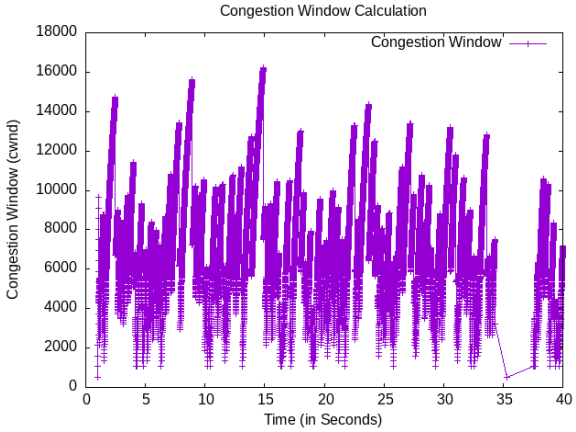
Application Data Rate: 1Mbps



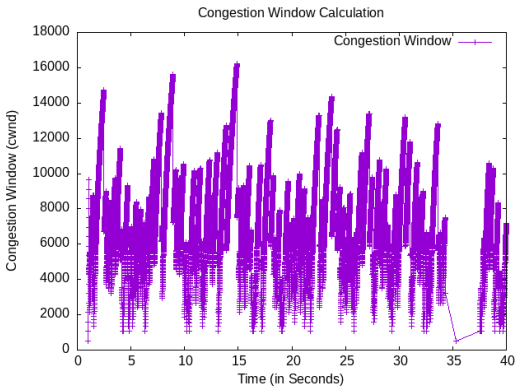
Application Data Rate: 2Mbps



Application Data Rate: 4Mbps



Application Data Rate: 8Mbps



Application Data Rate: 12Mbps

Question: How does application data rate affect congestion window size?

Answer: When the application data rate is more than or equal to the channel data rate there is no difference in the graph with respect to change in application data rate (in case of Application Data Rate = 4Mbps, 8Mbps, 12Mbps) as now channel data rate is the bottleneck. When the application data rate is less than channel data rate the packets are sent as a slower rate the rise and fall in congestion window size is less frequent and the number of packets send are also less.

Question: How does channel data rate affect congestion window size?

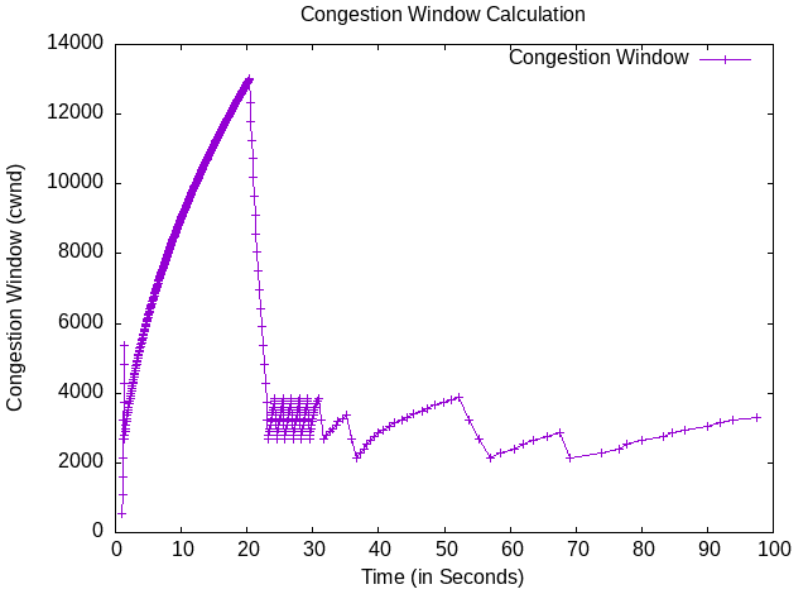
Answer: As the channel data rate increases the packets are send at faster rate and the acknowledgements from the reciever are also recieved at a faster rate thus the congestion window changes at faster rate. Thus we can see as the channel data rate is increased the graph is compressed along x-axis because of faster congestion window change.

Question: What relation do you observe between channel data rate and application data rate?

Answer: When the application data rate is more than or equal to channel data rate all the graphs will be identically same because the link can not carry anymore data and transport layer will have to wait and the rate at which application produces data makes no effect.
When the channel data rate is more than application data rate the rate of change of congestion window increases as channel data rate increses because the time taken to acknowledge a packet decreases.

Task 3:

1)



2) At x = 20, when UDP connection is started, there is a congestion because UDP clogs half capacity of N2-N3 link. So, the congestion window size is decreased continuously (till 2680) until there are no packet drop.
At 30 seconds, UDP clogs the whole capacity of N2-N3 link. At x = 36 there are multiple drop as a result the congestion window size further decreases from 2680 to 2144, now 2278 becomes new base value after every drop.