COL334 Assignment - 3

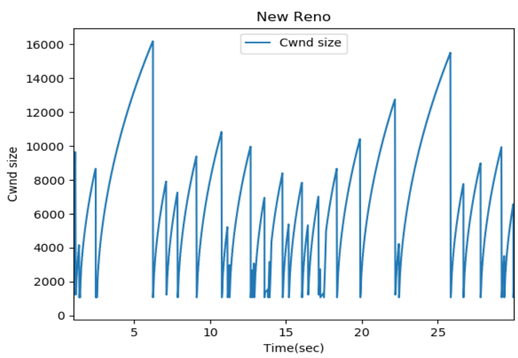
Aditya (2019CS50471)

25^{th} October, 2021

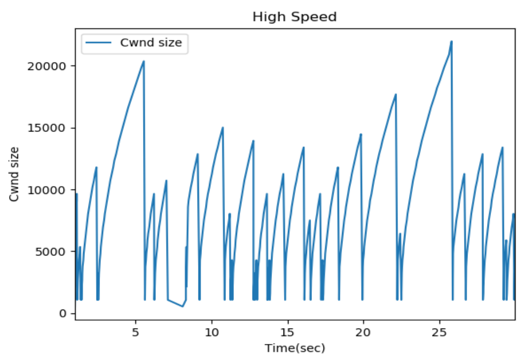
Contents

1 Congestion Window - Different Protocols

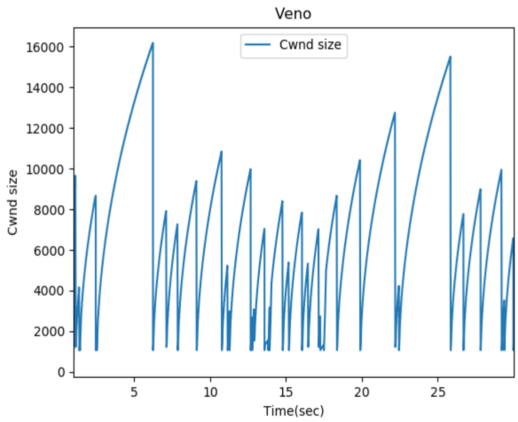
1.a Congestion Window vs Time plots



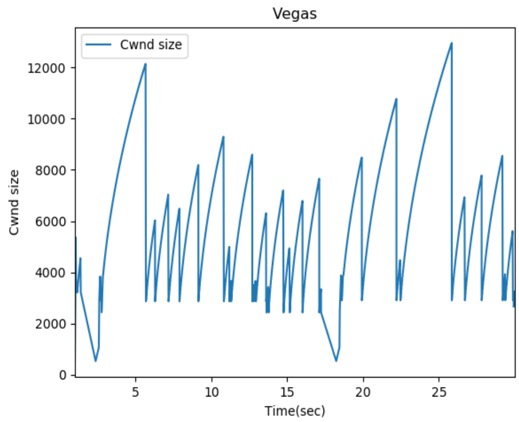
(a) Cwnd vs Time for NewReno protocol



(b) Cwnd vs Time for HighSpeed protocol



(a) Cwnd vs Time for Veno protocol



(b) Cwnd vs Time for Vegas protocol

1.b No of Dropped Packets

1. No of Dropped packets in NewReno protocol: 38
2. No of Dropped packets in HishSpeed protocol: 38
3. No of Dropped packets in Veno protocol: 38
4. No of Dropped packets in Vegas protocol: 39

No of dropped packets for all the protocols is almost same. The reason is that other parameters such as Channel delay, channel data rate, application data rate, packet size are constant. So, even if the protocols are different, no of dropped packets is almost same.

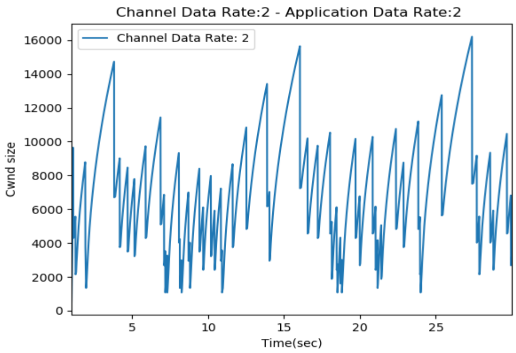
1.c Observations

1. **NewReno Protocol:** From the graph, we can see that when there is congestion in the network, then the congestion window is dropped to a low value. In slow start phase, cwnd = cwnd + SegmentSize and in congestion avoidance phase, cwnd = cwnd + max(1, $SegmentSize^2$/cwnd).
2. **HighSpeed Protocol:** Slow start phase of Highspeed protocol is same as slow start phase of standard TCP. But in congestion avoidance phase, cwnd varies as cwnd = cwnd + f(cwnd)/cwnd. Here f(cwnd) is a function of congestion window (cwnd). As cwnd increases, f(cwnd) increases. So, cwnd will grow faster than standard TCP, which can be seen from the plot. The plot is moving to higher values quickly than TCP New Reno and the peak is greater than TCP New Reno.
3. **Veno Protocol:** This protocol decides about the congestion in the network by monitoring network congestion level. This is a slight modification to Reno protocol (refines multiplicative decrease and linear increase algorithms). So, the plot of this protocol is almost similar to New Reno protocol.
4. **Vegas Protocol:** This protocol uses packet delay to identify congestion in the network unlike New reno which uses packet loss as an indicator for congestion. So, it detects congestion faster than other protocols, so the maximum cwnd size will be less than other protocols. This can be observed from the plot in which maximum cwnd size of Vegas is less than max cwnd size of other protocols.

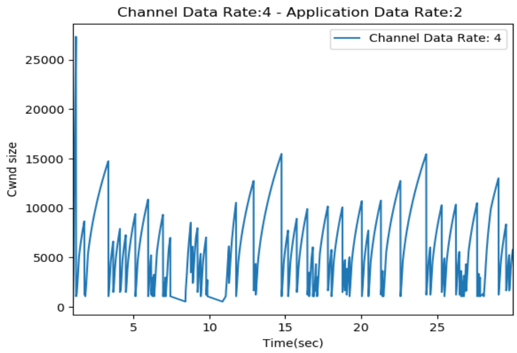
2 Variable Data Rate and Application Rate

2.a Congestion Window vs Time for variable data rate

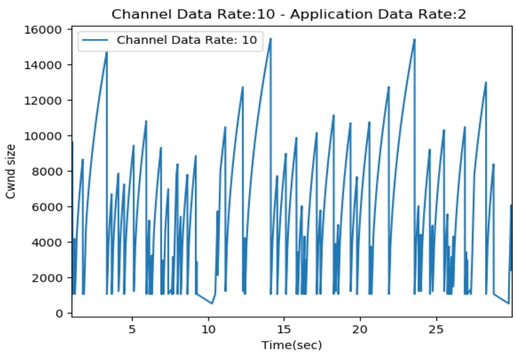
Application Data rate is 2Mbps in all these cases.



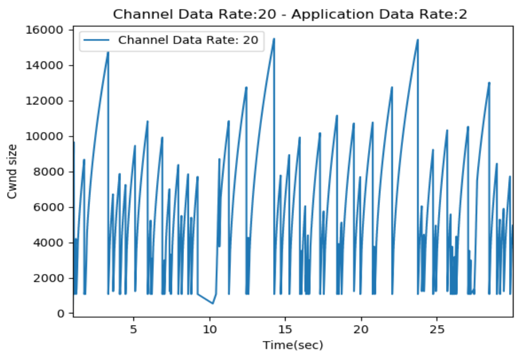
(a) Cwnd vs Time for Channel Data Rate - 2Mbps



(b) Cwnd vs Time for Channel Data Rate - 4Mbps



(a) Cwnd vs Time for Channel Data Rate - 10Mbps



(b) Cwnd vs Time for Channel Data Rate - 20Mbps

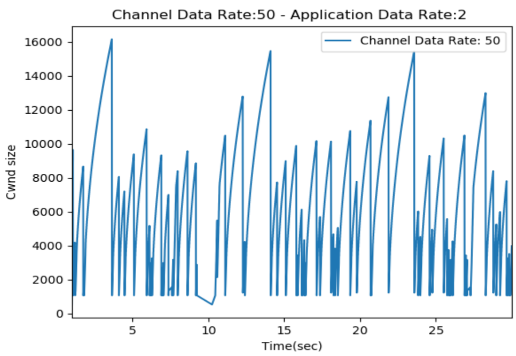
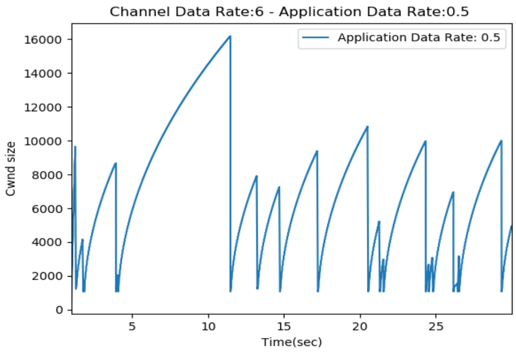


Figure 5: Cwnd vs Time for Channel Data Rate - 50Mbps

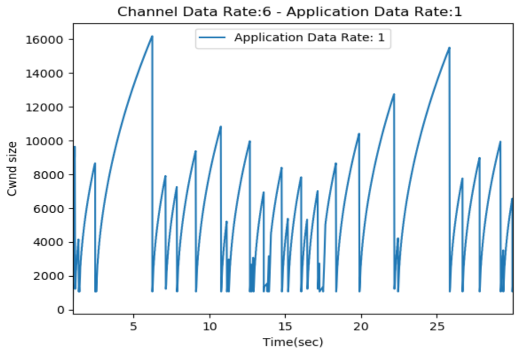
Here, we can see that all the plots are almost same i.e, if Application data rate is constant, channel data rate has almost no effect on congestion window. The reason is that Application is sending data at a constant rate of 2Mbps and channel data rate is greater than 2 Mbps in all the cases. So even if the channel data rate varies, as application data rate is constant size of congestion window will be same. So, here all the plots are same.

2.b Congestion Window vs Time for variable application rate

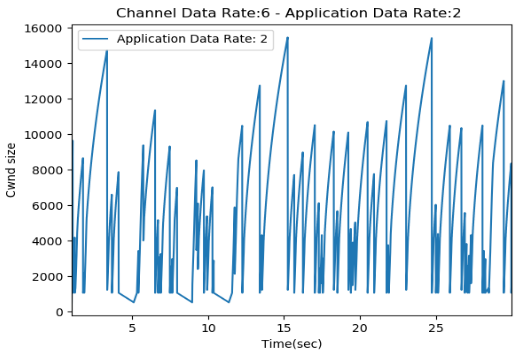
Channel Data Rate is 6Mbps in all these cases.



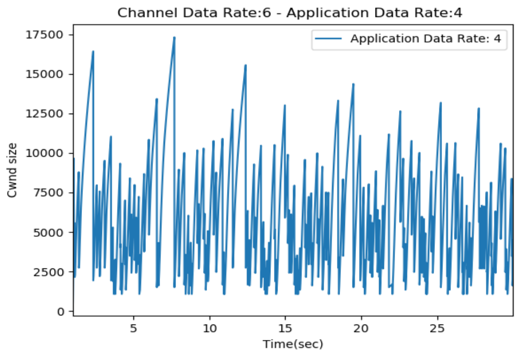
(a) Cwnd vs Time for Application Data Rate - 0.5Mbps



(b) Cwnd vs Time for Application Data Rate - 1Mbps



(a) Cwnd vs Time for Application Data Rate - 2Mbps



(b) Cwnd vs Time for Application Data Rate - 4Mbps

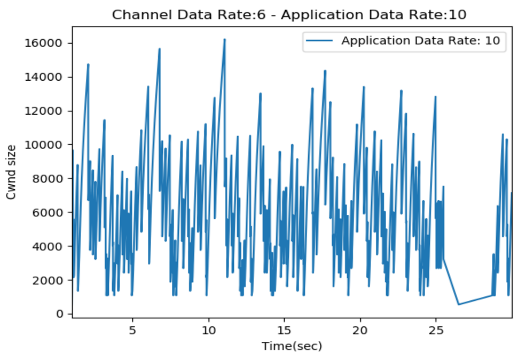


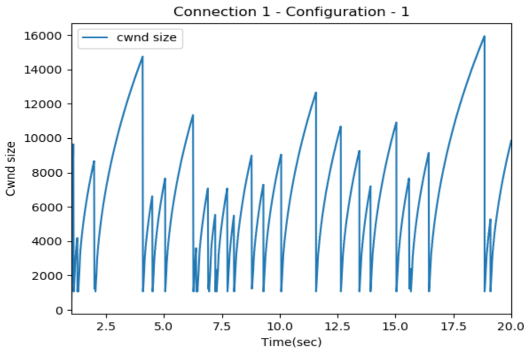
Figure 8: Cwnd vs Time for Application Data Rate - 10Mbps

Here, channel data rate is constant 6Mbps. So, if application data rate is low, channel can transmit data with less frequent packet loss. As application data rate increases and since channel data rate is constant, packet loss will occur more frequently. This trend can be observed in the graphs. As application data rate is increasing, pakcets get dropped frequently, so graph congestion window will drop more frequently.

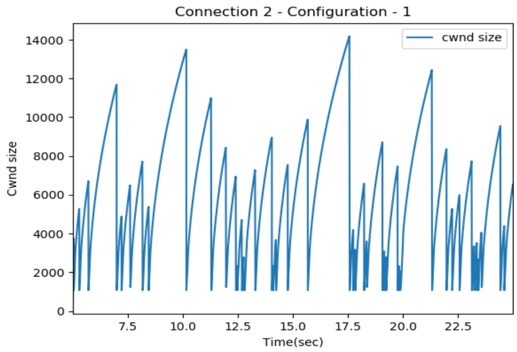
3 New Congestion Control Protocol

3.a Configuration - 1

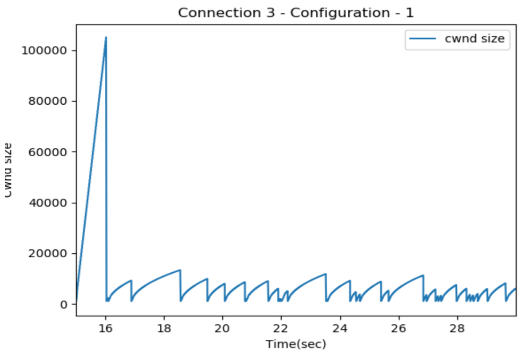
Plots:



(a) Configuration - 1, Connection - 1



(b) Configuration - 1, Connection - 2

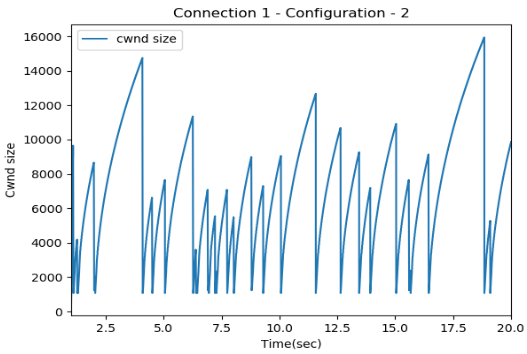


(a) Configuration - 1, Connection - 3

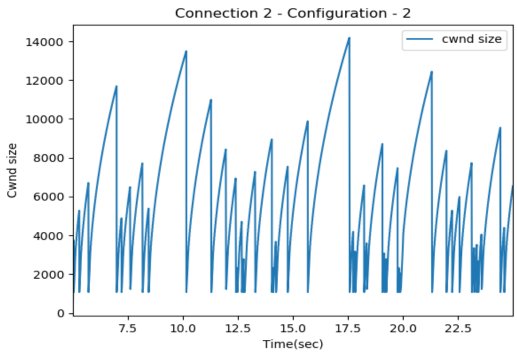
No of dropped Packets: 113

3.b Configuration - 2

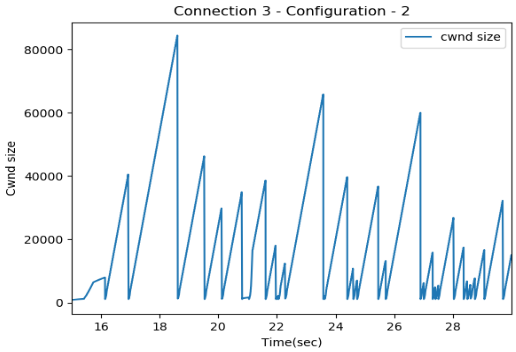
Plots:



(a) Configuration - 2, Connection - 1



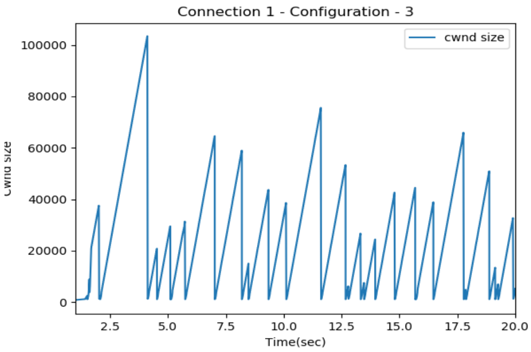
(b) Configuration - 2, Connection - 2



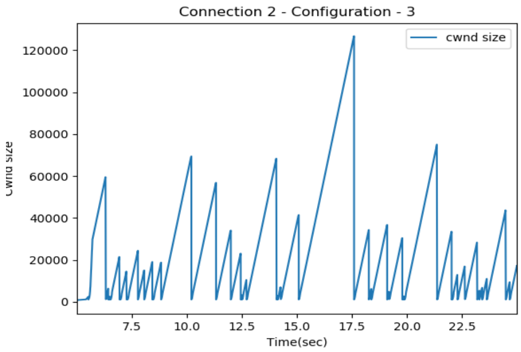
(a) Configuration - 2, Connection - 3

No of dropped Packets: 112

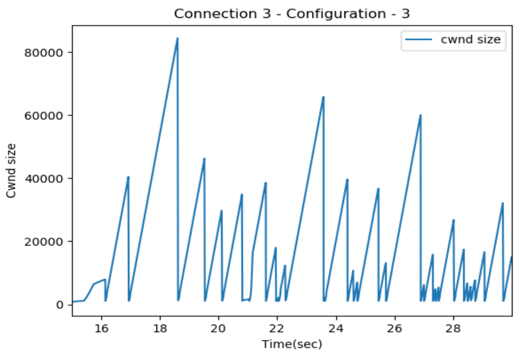
3.c Configuration - 3



(a) Configuration - 3, Connection - 1



(b) Configuration - 3, Connection - 2



(a) Configuration - 3, Connection - 3

No of dropped Packets: 110

3.d Trends and Observations

1. For connection - 1 and connection - 2, we used TCPNewReno in configurations 1 and 2, TCPNewRenoCSE in configuration - 3 whereas for connection - 3, we used TCPNewReno in configuration - 1 and TCPNewRenoCSE in configurations 2 and 3

2. In congestion avoidance phase of NewRenoCSE, cwnd is varying as cwnd = cwnd + 0.5*SegmentSize whereas in Congestion Avoidance phase of TCPNewReno, cwnd is varying as cwnd = cwnd + max(1, $SegmentSize^2$/cwnd)

3. These trends can be observed from the graph. In congestion avoidance phase of NewRenoCSE, cwnd is varying linearly, but for TcpNewReno it is varying according to the function given above.

4 Appendix

1. First.cc - Takes protocol name as input and generates a csv file with time, old cwnd, new cwnd and prints no of dropped packets.

2. part1.py - Once files for all the 4 protocols are generated using above file (First.cc), this file generates Congestion window vs time plots for all the 4 protocols and saves them.

3. Second.cc - Takes Channel Data rate, application data rate as input, generates a csv file with time, old cwnd, new cwnd.

4. part2.py - Once files for all the 10 different pairs of channel data rate, application data rate are generated using above file (Second.cc), this file generates Congestion window vs time plots for all the 10 pairs.

5. Third.cc - Takes configuration (1/2/3) as input and outputs csv file containing Time, old cwnd, new cwnd corresponding to 3 connections and prints no of dropped packets.

6. part3.py - Once files for all the 3 configurations are generated using above file (Third.cc), this file generates Congestion window vs time plots for all the 3 connections for 3 configurations and saves them.

7. TcpNewRenoCSE.cc, TcpNewRenoCSE.h - New congestion model files

8. For graph plotting, I used matplotlib and pandas libraries.