Assignment 3 On Network Simulator (ns3)

Done By: Mohit Sharma (2019CS10372)

Note: Place the submission folder inside **ns-allinone-3.29/ns-3.29/** best experience.

For Part 1 and Part 2, code have been reused from sixth.cc file in examples/tutorials of ns3.

Part3 code for third.cc is also written taking reference of sixth.cc.

Part3 NewRenoCSE have been made by inheriting NewReno.

move.sh : moves required file to scratch folder and compiles them.

Run: **./move.sh**

Part1: Files Inside Q1

(Other than below given files there are 4 plots file also**)**

1) **first.cc** : Generates changes in congestion window with time using different protocols in "first.cwnd" file. Place it in scratch folder and compile by: ./waf
Run: **./waf --run "scratch/first.cc --protocol=<ProtocolName>"**
(In place of <ProtocolName> use NewReno, HighSpeed, etc.)

2) **plot.plt** : Plots congestion window size vs time from "first.cwnd" file in "<ProtocolName>_plot.png" file.
Run: **gnuplot -e "filename='<ProtocolName>'" plot.plt**

3) **run.sh** : Runs the first.cc code and generates the required plot. Takes protocol name as arg.
Run: **./run.sh <ProtocolName>**

4) **part1.sh** : Run first.cc and generates plots for all protocol given in the assignment.
Run: **./part1.sh**

5) **moveToScratch.sh** : moves first.cc to scratch folder and compiles it.
Run: **./moveToScratch.sh**

Part1: Plots and Inferences

1) Packet Drop Count:

NewReno : 38

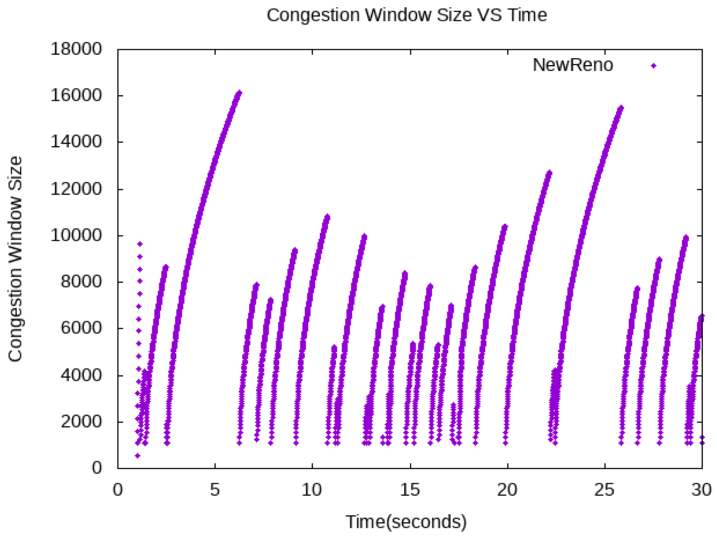
HighSpeed : 38

Veno : 38

Vegas : 39

2) Plots:

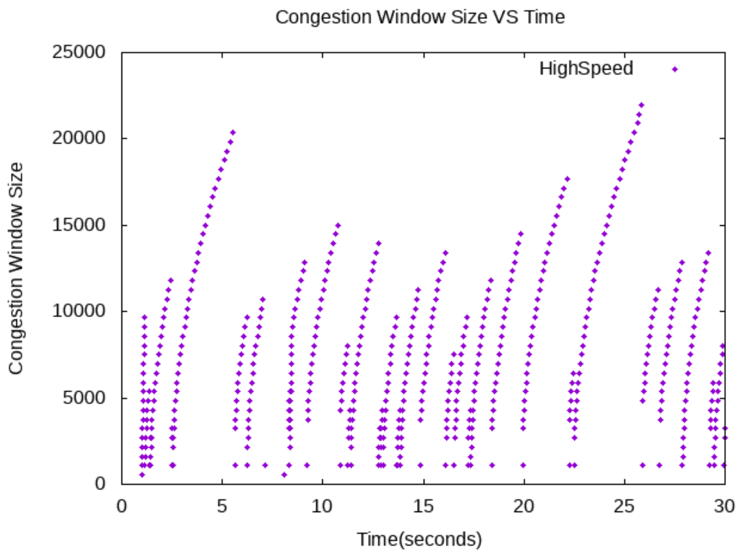
1) NewReno



cwnd += min(N,SMSS), N = previously unacknowledged bytes, SMSS = threshold bytes

In the slow start phase the congestion window is increased by the threshold value set or if previous unacknowledged bytes is lesser that it is used. When congestion window exceeds the threshold value then congestion avoidance phase starts. In the congestion avoidance phase the window size is incremented only by 1 full size segment. That is why the tip of the curves start bending.

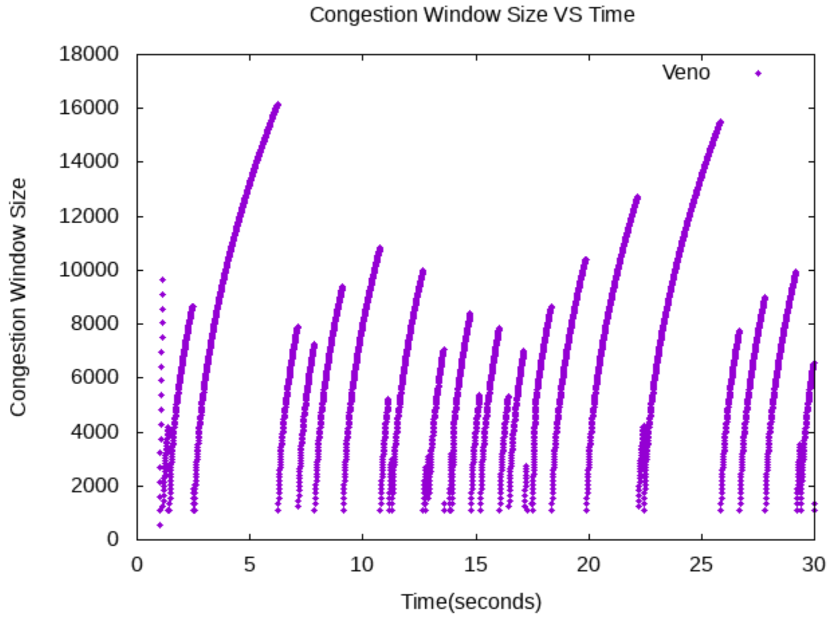
2) HighSpeed



In highspeed protocol large congestion window sizes are allowed(It goes beyond 20000 in the graph, while for NewReno it was around 16000). The congestion window increases with following equation:

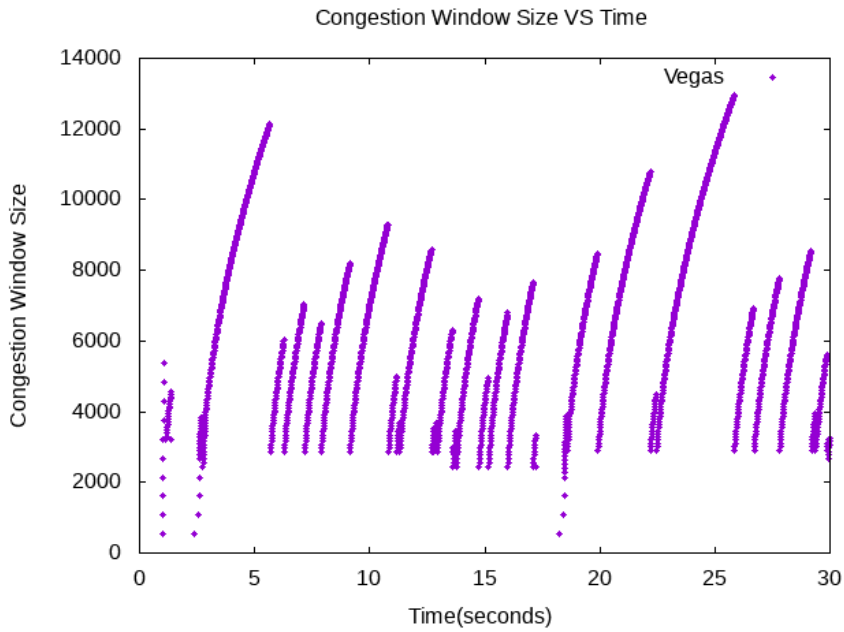
cwnd = cwnd + a(cwnd)/cwnd, here a() is a lookup function. While a(cwnd) evaluates to 1 the protocol behaves like NewReno. The congestion window increases rapidly here as less acknowledgments are required at higher window size to increase cwnd. The density of packets throw is also low.

3) Veno



The veno protocol uses almost the same congestion avoidance and slow rate algorithms as new reno and hence the graph is same. The difference in the protocols is that in veno the cwnd size is cut by 1/5 times when random loss due to bit errors are likely to have happened. It actually uses Vegas method to decide if it is in congestion state or not.

4) Vegas



In vegas protocol we can observe that there is a minimum kind of congestion window size, i.e, the size is increase to that value very quickly. The packet drops are tried to be prevented by maintaining a backlog at the bottleneck queue. The algorithm is turned off and on in the course of its working.

Part2: Files Inside Q2

(Other than below given files there are 10 plots file also**)**

1) **second_1.cc** : Writes congestion window change at different channel rate in “second_1.cwnd” file
   Run: **./waf --run “scratch/second_1 --rate=<DataRate>”**

2) **second_2.cc** : Writes congestion window change at different application rate in “second_2.cwnd”
   Run: **./waf --run “scratch/second_2 --rate=<DataRate>”**

3) **plot1.plt** : Plots congestion window size vs time from “second_1.cwnd” file in “parta_<rate>_plot.png” file.
   Run: **gnuplot -e "filename='<DataRate>'" plot1.plt**

4) **plot2.plt** : Plots congestion window size vs time from “second_2.cwnd” file in “partb_<rate>_plot.png” file.
   Run: **gnuplot -e "filename='<DataRate>'" plot1.plt**

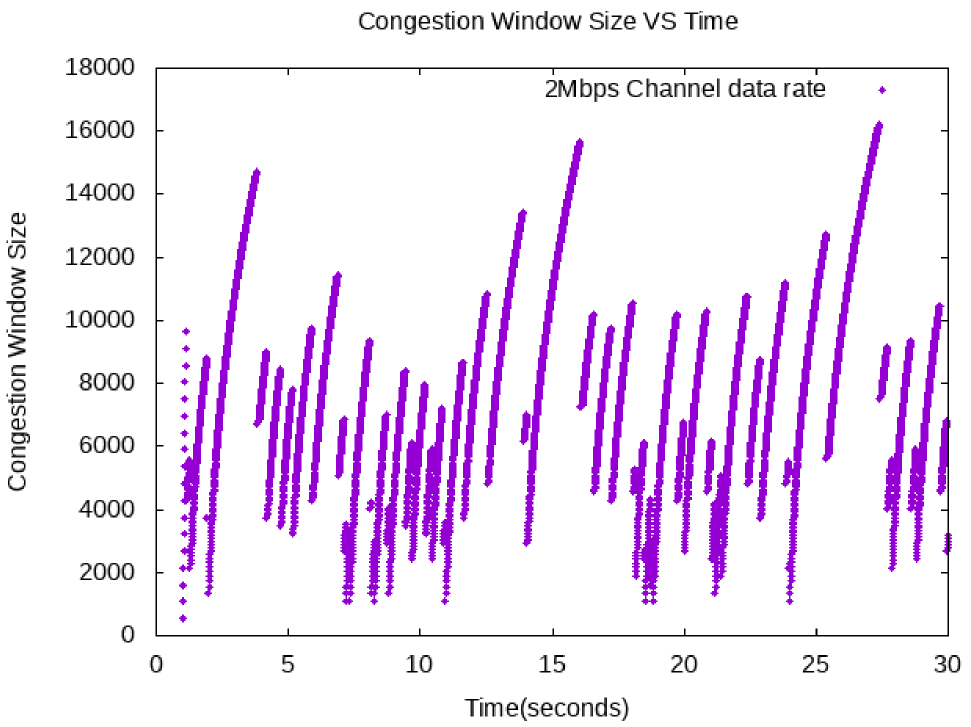
5) **run.sh** : Runs “second_1.cc” or “second_2.cc” code and generates the corresponding plot. Takes two arguments, First one should be 1 or 2 corresponding to part a(changing channel data rate) or b(changing application data rate), second parameter should be the corresponding data rate.
   Run: **./run.sh <Part1,2> <DataRate>**

6) **part2.sh** : Runs code to create all required plots in the assignment
   Run: **./part2.sh**

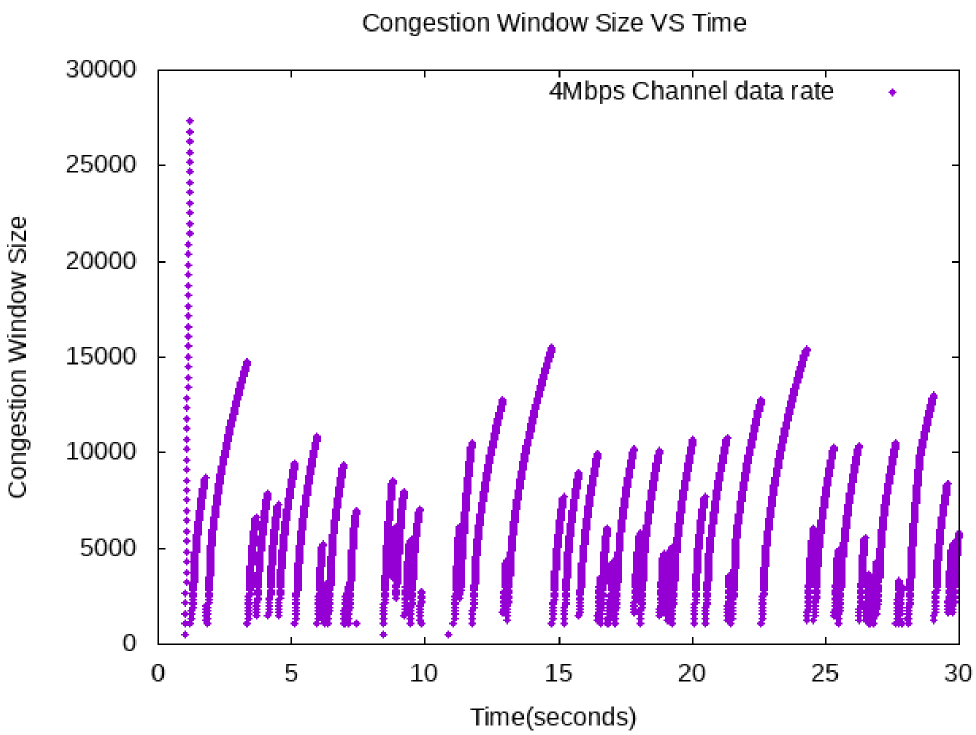
7) **moveToScratch.sh** : moves second_1.cc and second_2.cc to scratch folder and compiles it.
   Run: **./moveToScratch.sh**

Part2: Plots and Inferences

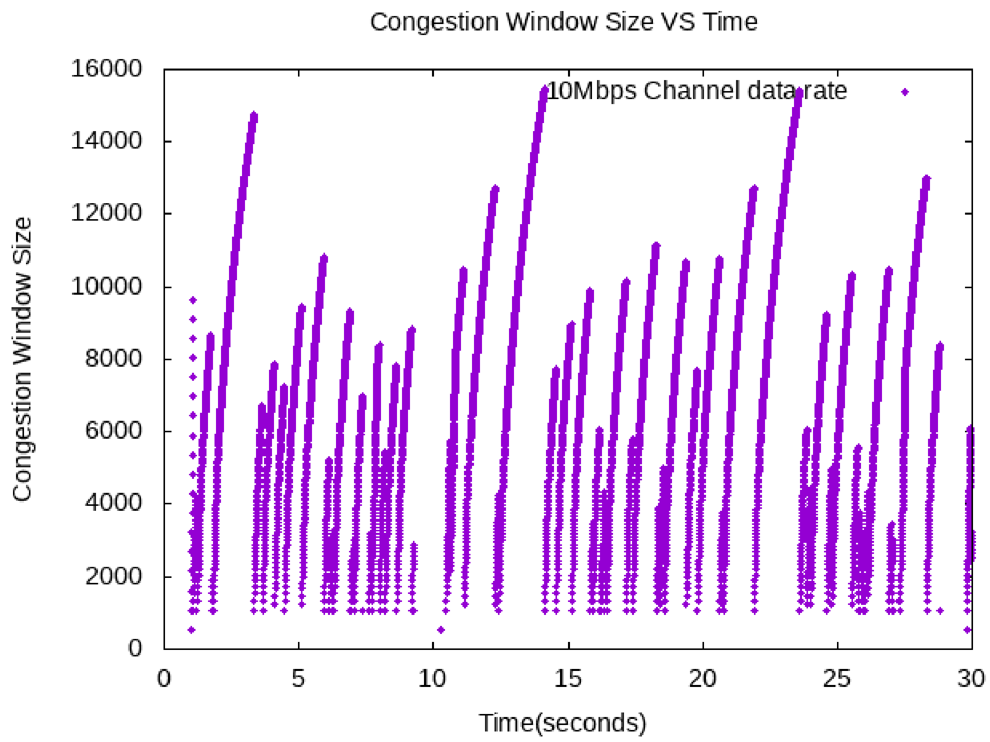
a-1) Channel data rate of 2Mbps



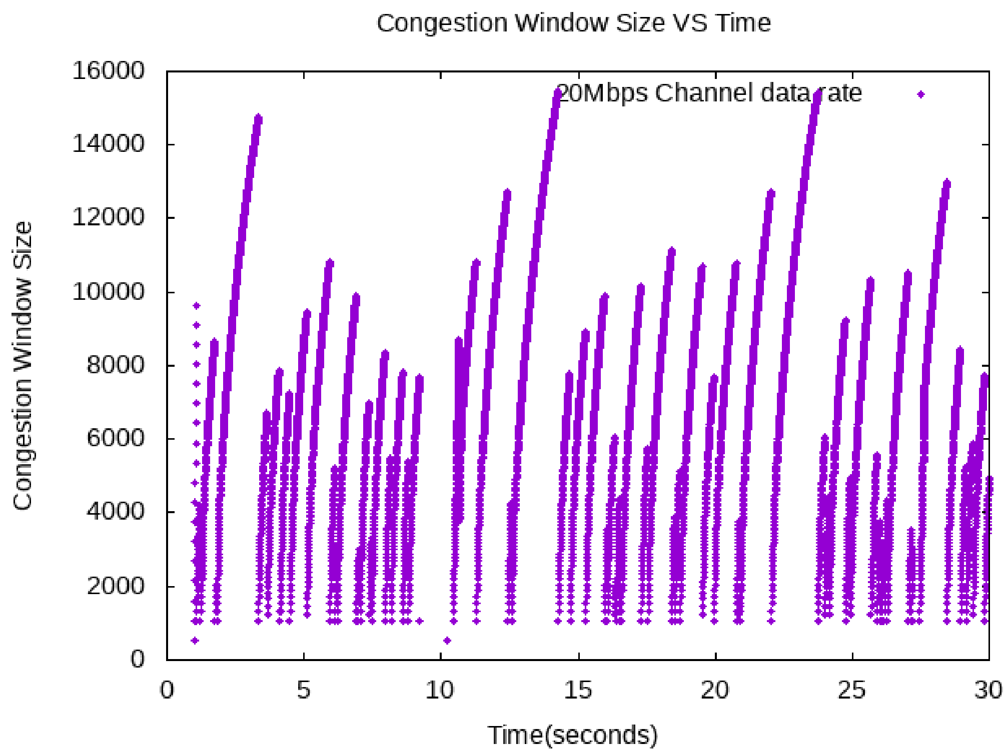
a-2) Channel data rate of 4Mbps



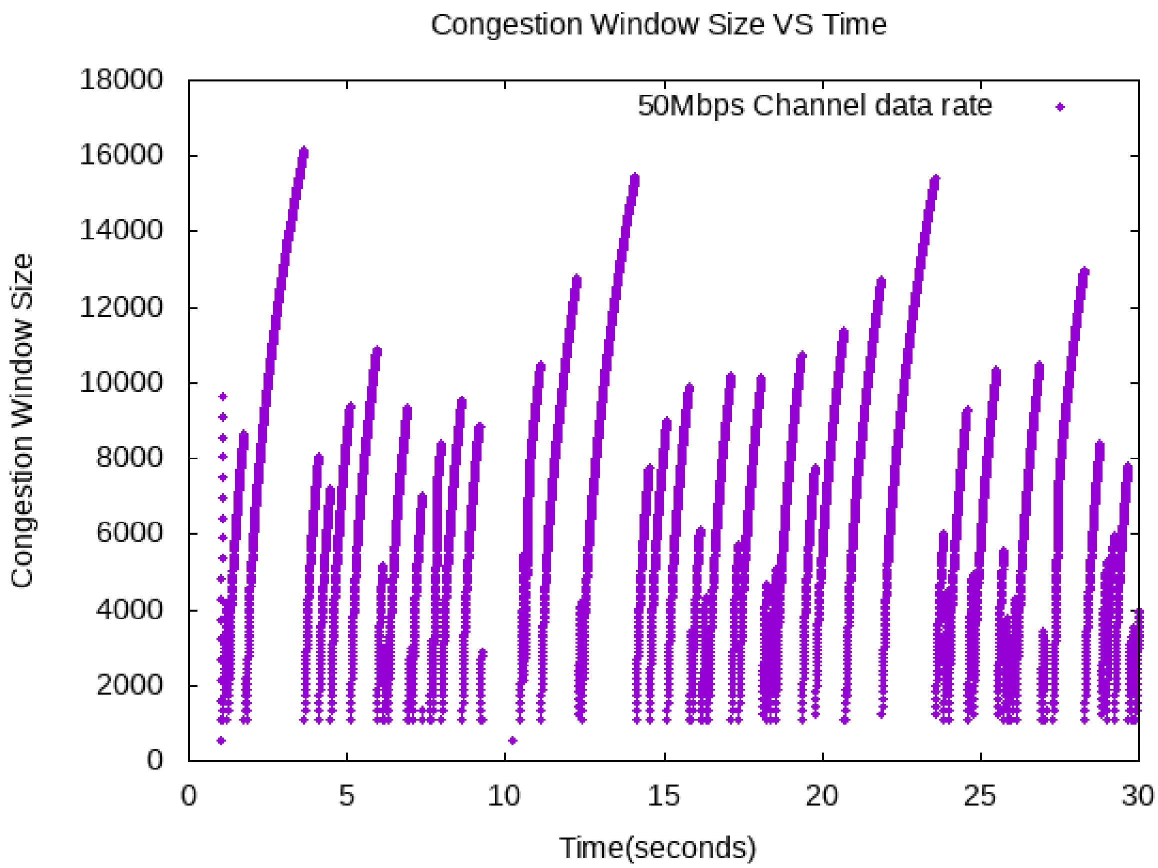
a-3) Channel data rate of 10Mbps



a-4) Channel data rate of 20Mbps



a-5) Channel data rate of 50Mbps

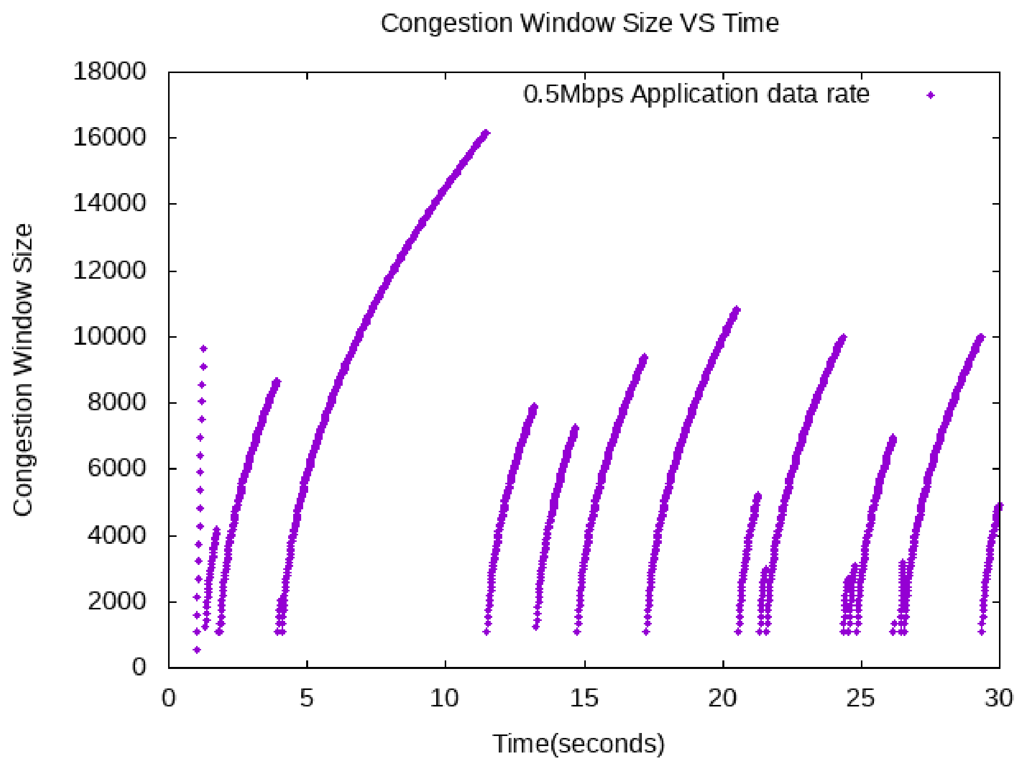


Observations:

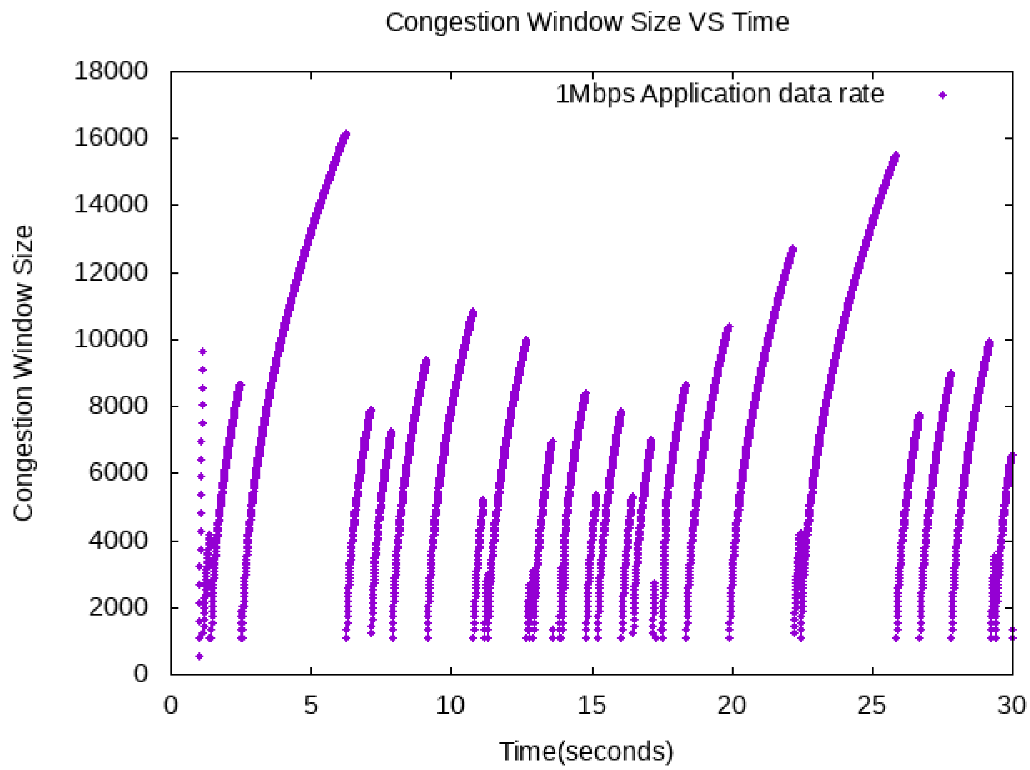
For very small channel data rate of 2Mbps (that is the one <= application data rate), the congestion window size is dropped to half from local maximas in graph on congestion detection which is not the case in other plots.

As channel data rate increases from 2mbps to 4 mbps the number of peaks increases and after that there is no significant increase in number of peaks. This is because the application rate is also 2Mbps and earlier the data send by application was at same rate sent through channel, but on increase in channel data rate there is more congestion of as sink does not receive them as fast as channel sends, and hence this congestion increases packets dropped.

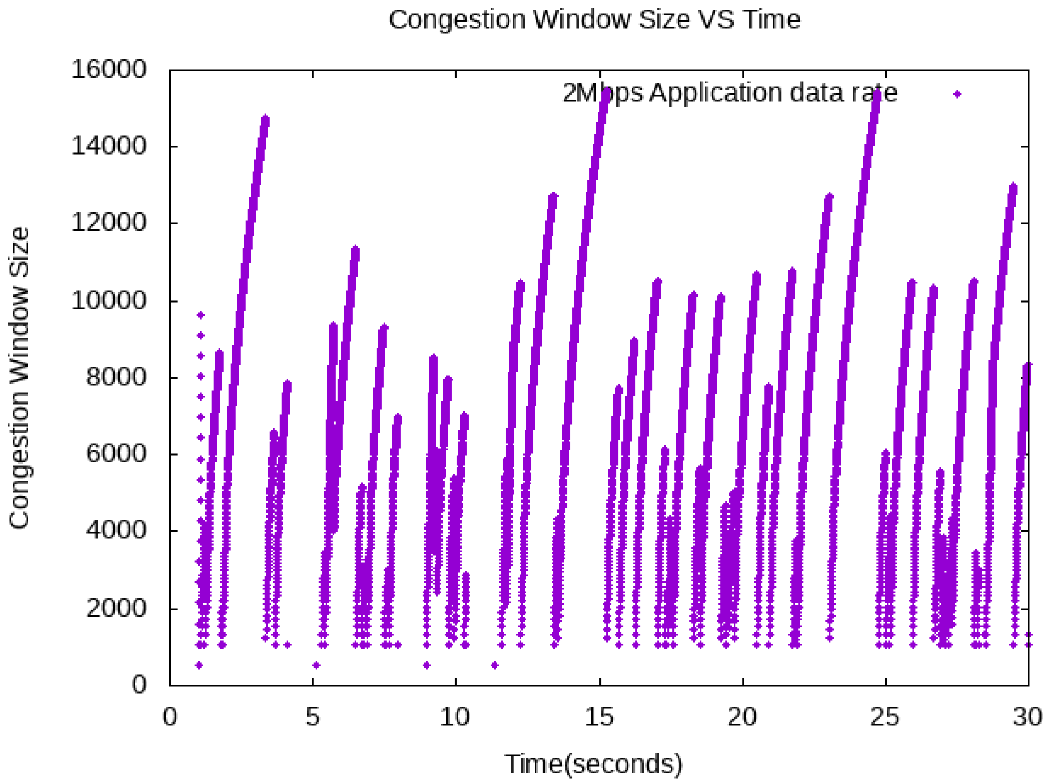
b-1) Application data rate of 0.5Mbps



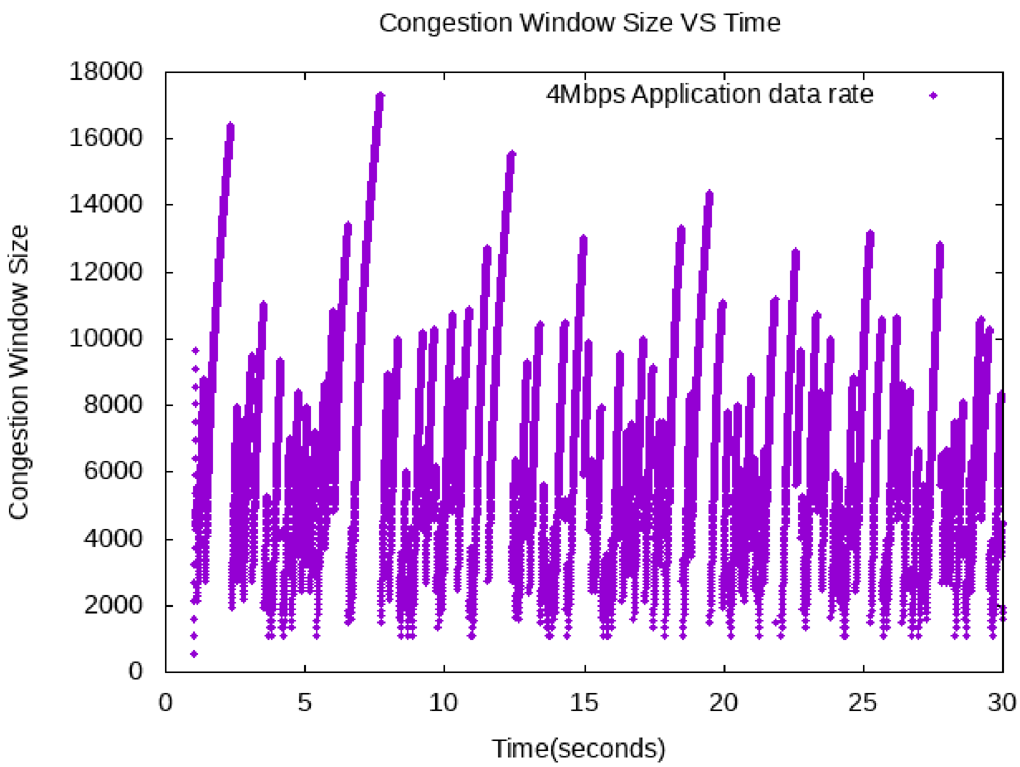
b-2) Application data rate of 1Mbps



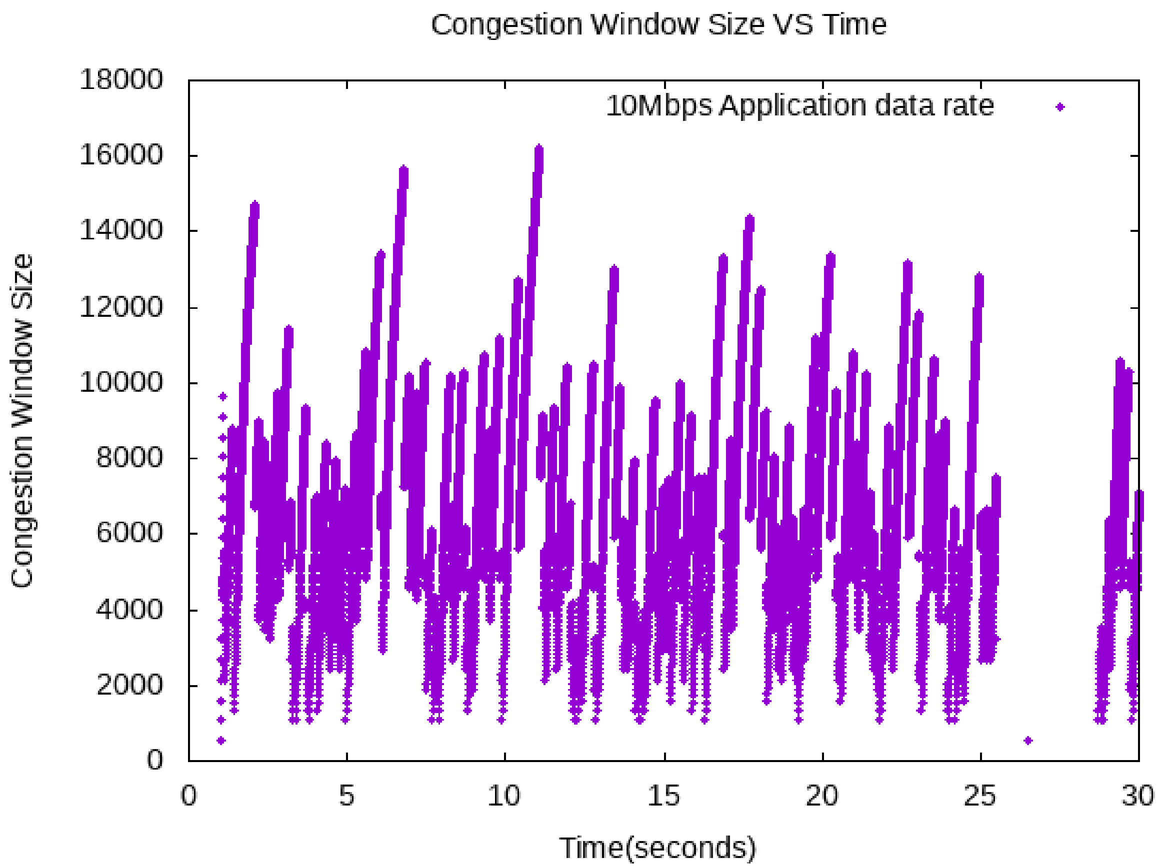
b-3) Application data rate of 2Mbps



b-4) Application data rate of 4Mbps



b-5) Application data rate of 10Mbps



Observations:

We can observe that the number of packets dropped would increase with increase in application data rate. This can be inferred from the large number of peaks in the graph at higher application rate. As we send packets with larger speed the congestion increases and packets are to be dropped.

Part 3: Files inside Q3

(Other than below given files, there are 9 plots also)

1) **third.cc** : Writes congestion window changes in three files, third_1.cwnd, third_2.cwnd, third_3.cwnd corresponding to each connection. Takes configuration as input (number (1,2 or 3)).

    Run: **./waf --run "third.cc --conf=<configuration Number>"**

2) **plot.py** : Plots graph for congestion window vs time for given configuration and connection and reading the required file(filename is hardcoded as third_<connection>.cwnd).

    Run: **python3 plot.py <configuration> <connection>**

3) **TcpNewRenoCSE.cc** : As required (inherits TcpNewReno)

    Place it inside "ns-allinone-3.29/ns-3.29/src/internet/model"

4) **TcpNewRenoCSE.h** : As required (inherits TcpNewReno)

    Place it inside "ns-allinone-3.29/ns-3.29/src/internet/model"

    (To use file 3 and 4, we have to add edit "ns-allinone-3.29/ns-3.29/src/internet/wscript" accordingly)

5) **run.sh** : Runs "third.cc" code and generates 3 corresponding plots. Takes configuration as arg.

    Run: **./run.sh <configuration>**

6) **part3.sh** : Run code to generate all plots for the assignment.

    Run : **./part3.sh**

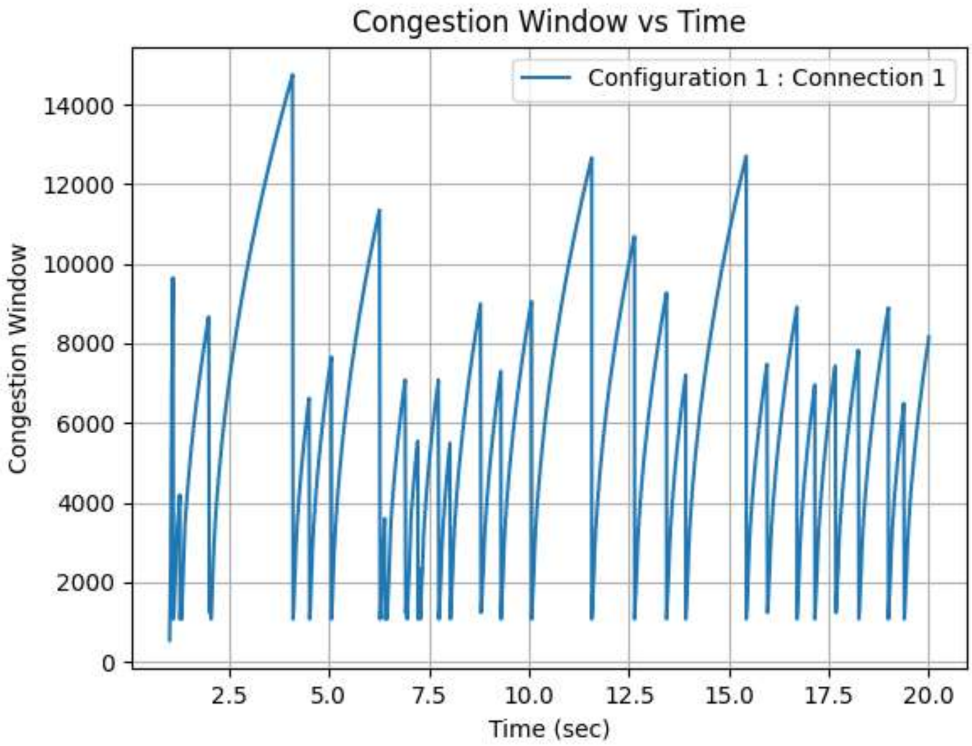
7) **moveToScratch.sh** : moves third.cc to scratch folder and compiles it.

    Run: **./moveToScratch.sh**

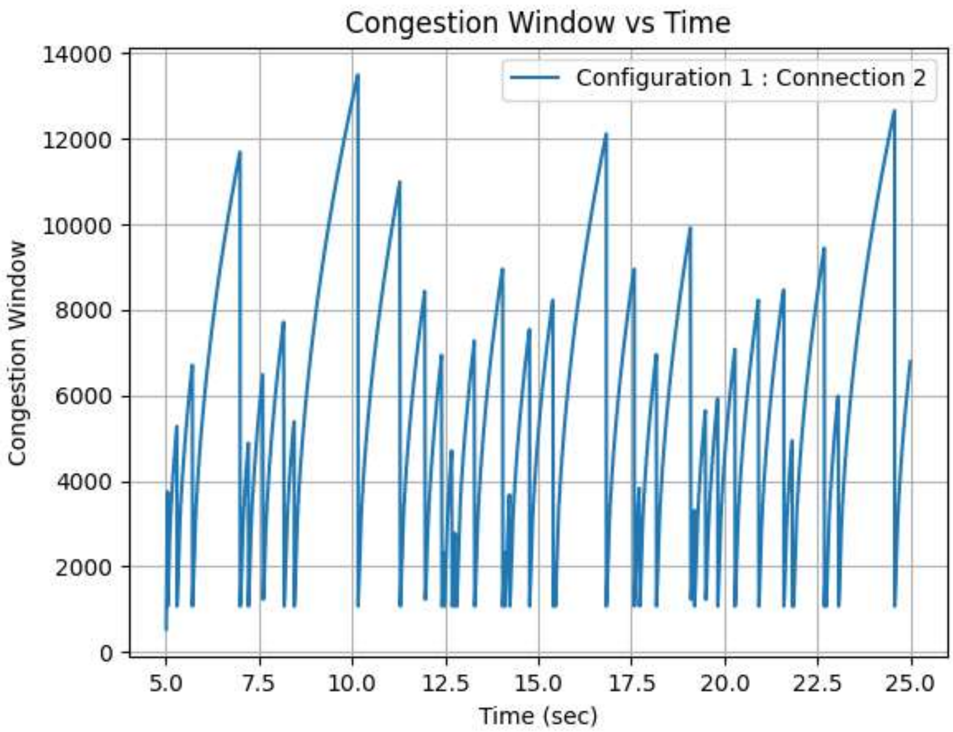
Part3 : Plots and Inferences

1) Configuration1

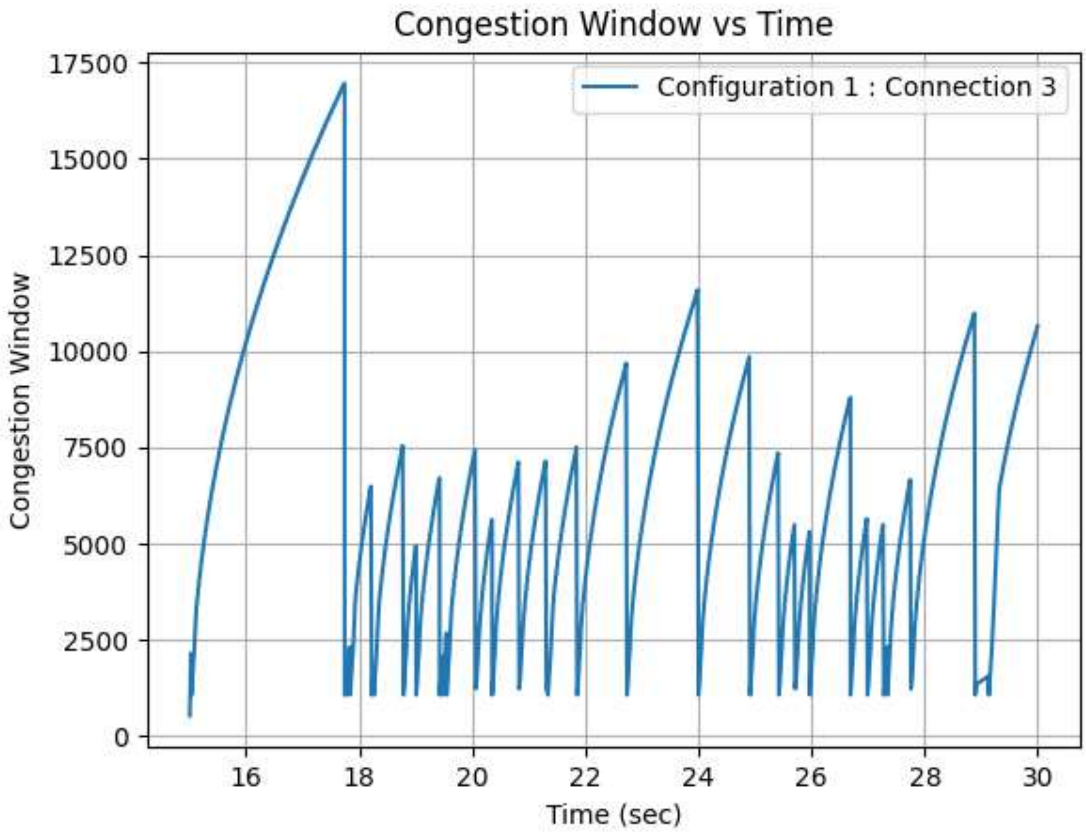
a) Connection 1



b) Connection 2



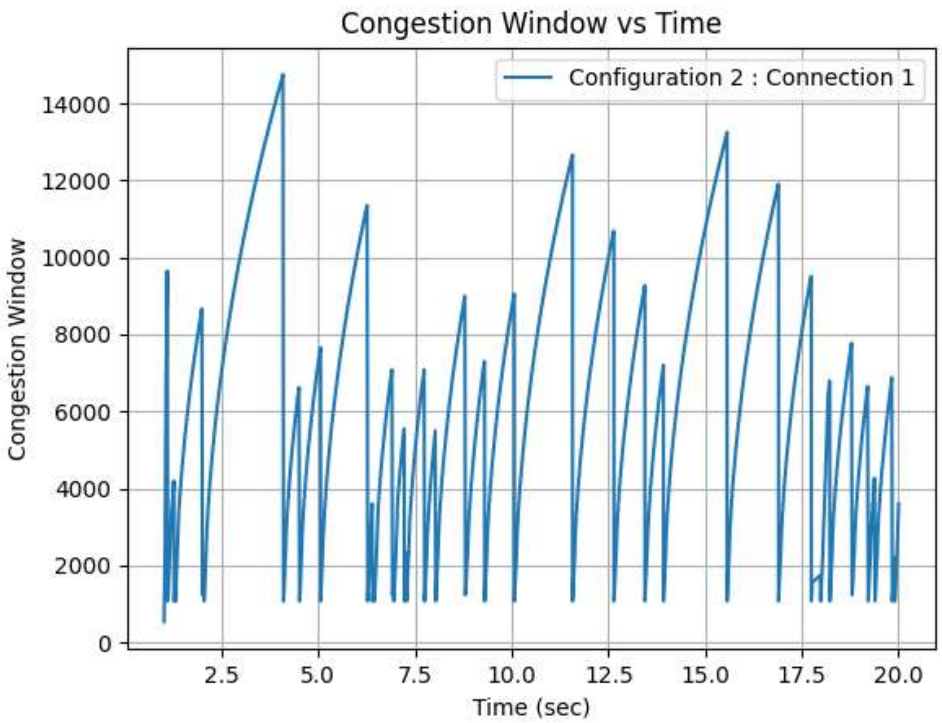
c) Connection 3



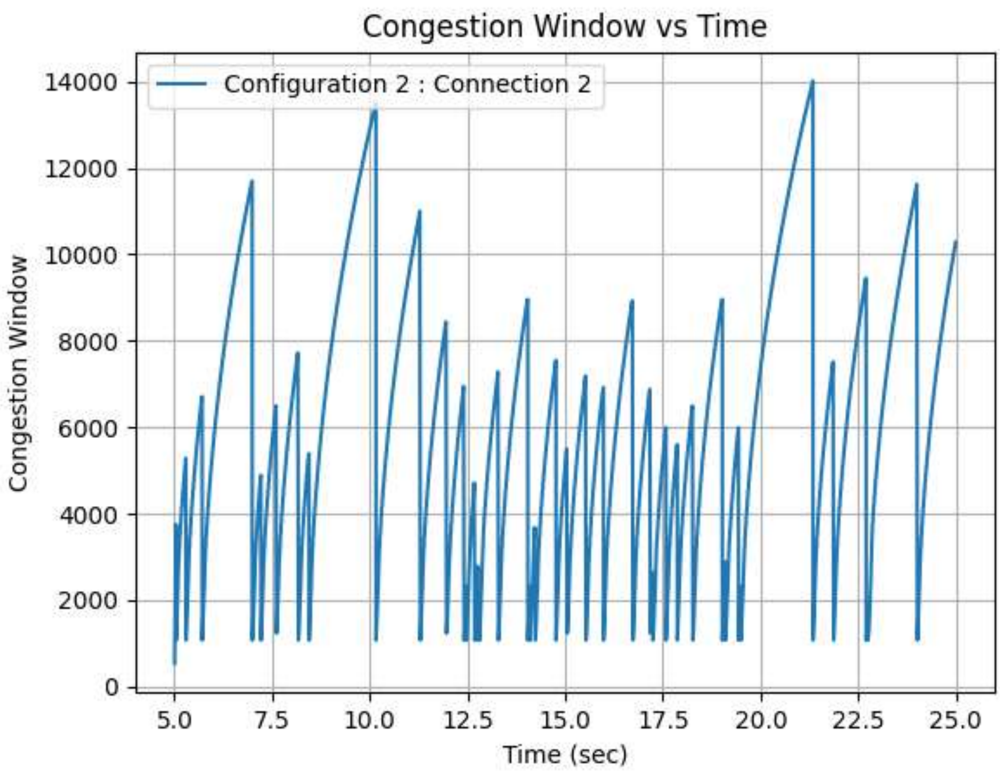
Total Packets dropped in Configuration 1: 106

2) Configuration2

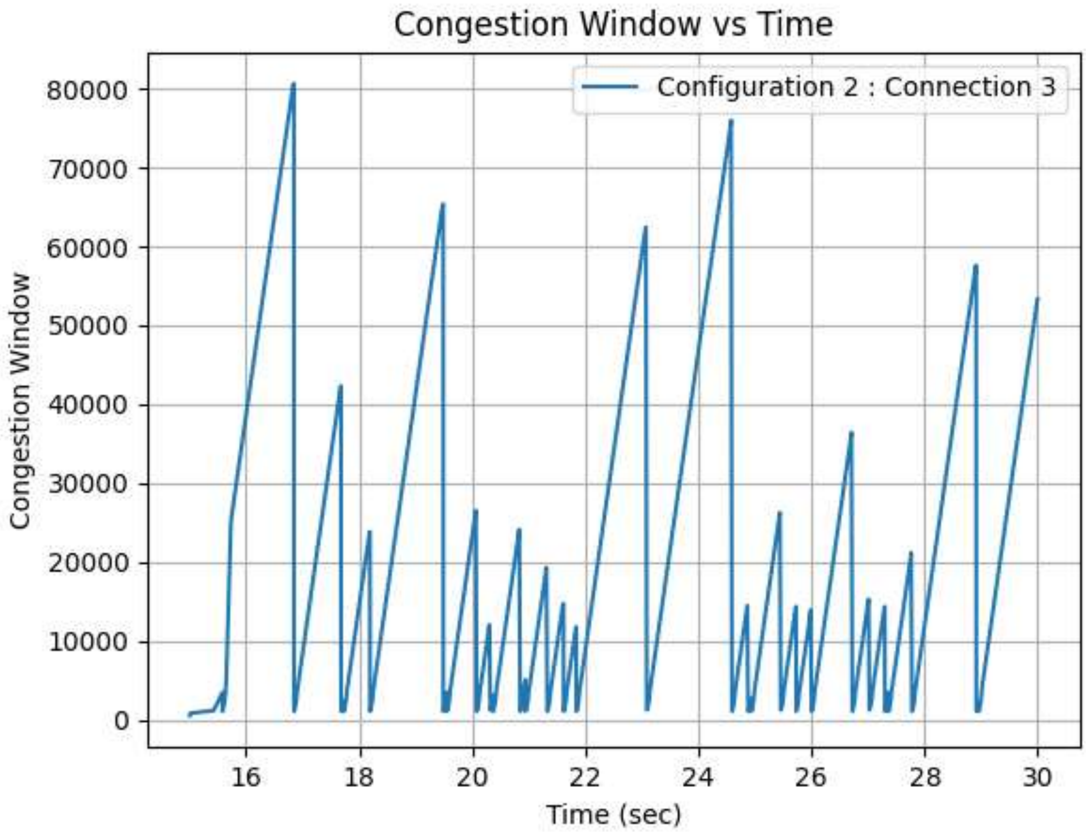
a) Connection 1



b) Connection 2



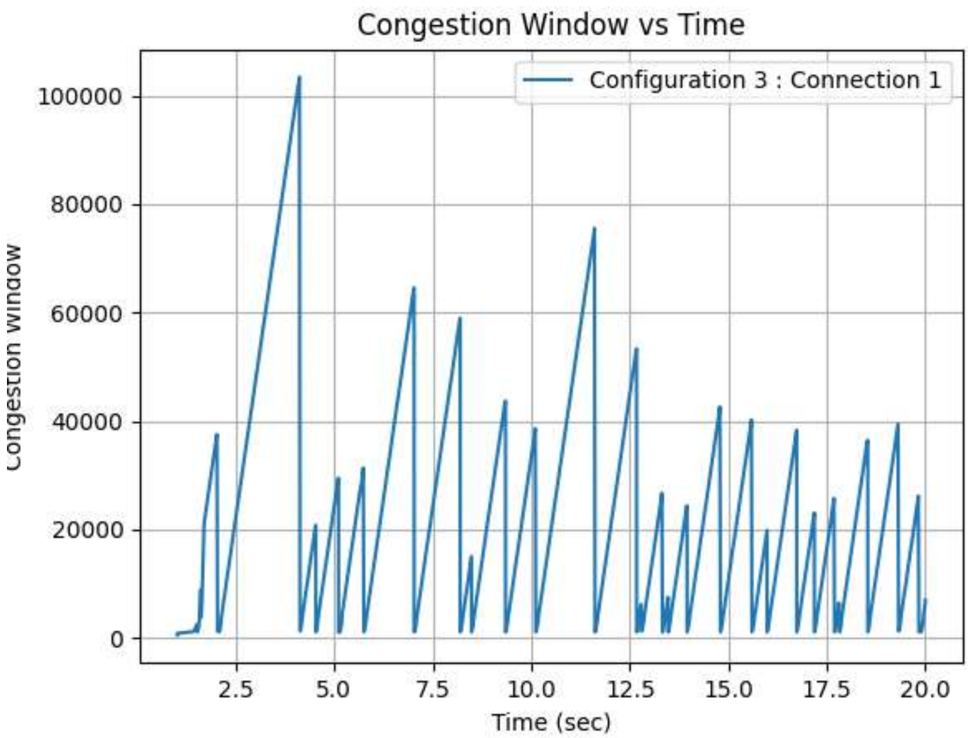
c) Connection 3



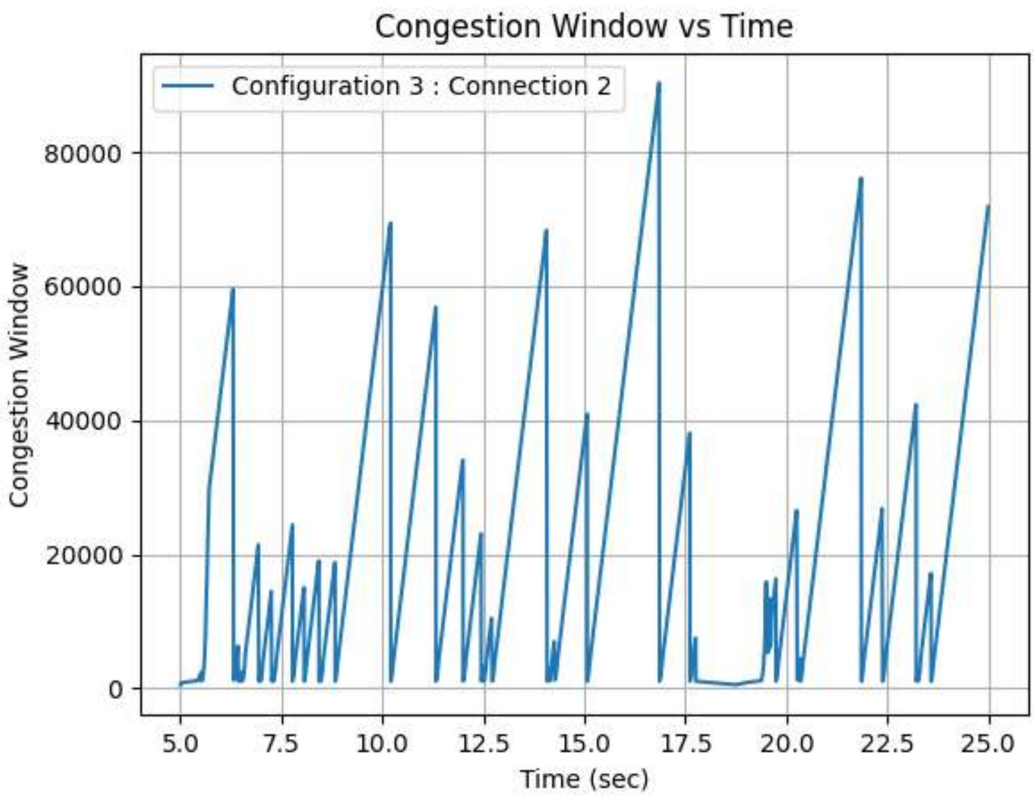
Total Packets dropped in Configuration 2: 110

3) Configuration3

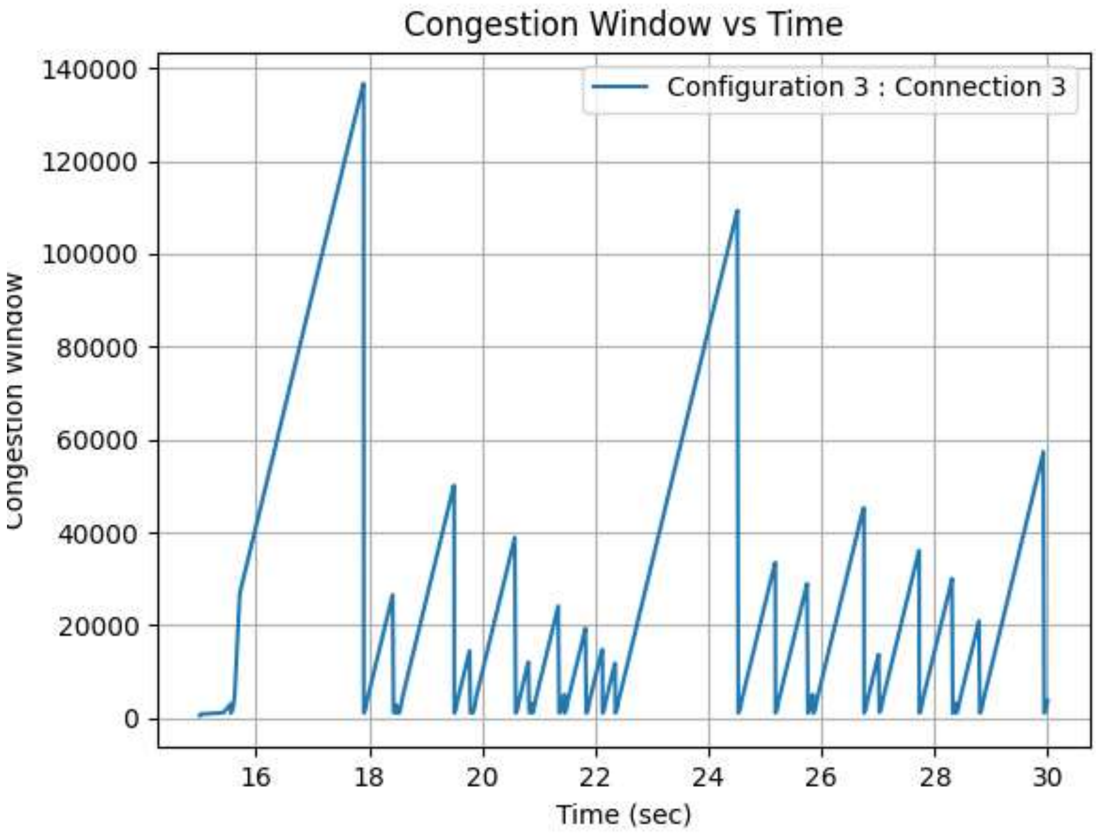
a) Connection 1



b) Connection 2



c) Connection 3



Total Packets Dropped in Configuration 3: 105

TcpNewRenoCSE increases congestion window rapidly. This is clearly visible in the graphs, in all configurations, any connection which uses TcpNewRenoCSE have peaks upto 80K, while the connections using TcpNewReno have peak upto 18K only.

Using TcpNewRenoCSE increases the congestion window to large size and thus allows for some extra packets in the queue to sender for some extra time, this is why the total number of packets drop in configuration 3.

When only connection 3 uses TcpNewRenoCSE, then even more number of packets get dropped in N1-N3 connections as it does not have a large congestion window.