CSE232 Monsoon 2024: PA04
Network simulation and TCP congestion control analysis using ns3
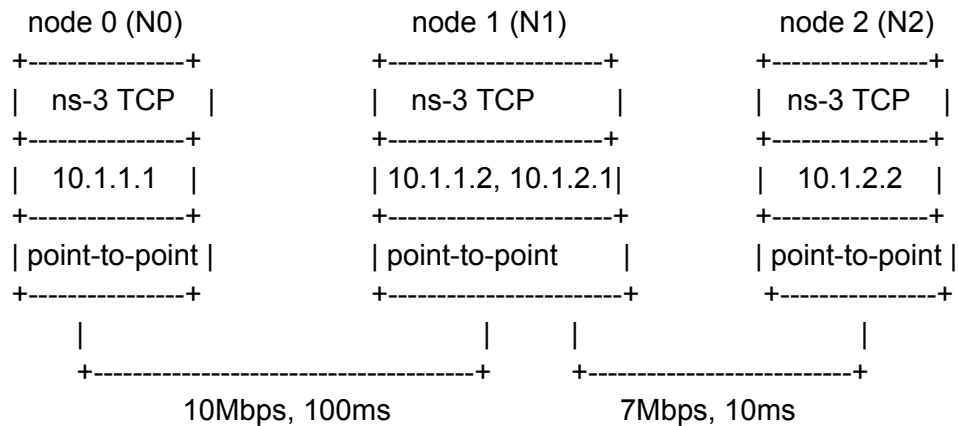## Due date: November 22, 2024
Total points: 18

The ns3 script for two network nodes, tcp-example.cc, is available here. Change this script to simulate a network of three nodes (say, N0, N1, and N2) with the following details.
Bandwidth between N0 & N1 = 10Mbps; delay = 100ms
Bandwidth between N1 & N2 = 7Mbps; delay = 10ms
Data is transferred over TCP on the link.

```
   node 0 (N0)                node 1 (N1)                node 2 (N2)
+----------------+        +----------------------+        +----------------+
|   ns-3 TCP    |        |    ns-3 TCP         |        |   ns-3 TCP    |
+----------------+        +----------------------+        +----------------+
|   10.1.1.1    |        | 10.1.1.2, 10.1.2.1|        |   10.1.2.2    |
+----------------+        +----------------------+        +----------------+
| point-to-point |        | point-to-point      |        | point-to-point |
+----------------+        +----------------------+        +----------------+
        |                       |       |                        |
   +-------------------------------------+     +--------------------------+
              10Mbps, 100ms                        7Mbps, 10ms
```

The first few lines in the main function in the script are as follows..
std::string tcp_variant = "TcpCubic";
std::string bandwidth = "5Mbps";
std::string delay = "5ms";
std::string queuesize = "10p";
double error_rate = 0.000001;
int simulation_time = 10; //seconds

The above code lines configure some of the parameters shown in the table below. The following table specifies the default parameter values that you should configure in the given script. You can change these values as needed in the exercise questions.

Parameter	Default value
Link bandwidth between the two nodes, N0-N1 & N1-N2	10Mbps, 7Mbps, respectively
One way delay of the link, N0-N1 & N1-N2	100ms, 10ms, respectively
Loss rate of packets on the link. This covers losses other than those that occur due to buffer drops at N1	0.000001

Queue size of the buffer at node 1	50 packets
TCP variant used	TcpNewReno
Simulation time	10 seconds
Application payload size	1460 bytes

Upon running successfully, the "tcp-example.cc" script produces the following output files.

- "Tcp-example-0-0.pcap", "tcp-example-1-0.pcap", and "tcp-example-2-0.pcap", which are the pcap logs collected at nodes N0, N1 and N2 respectively. The congestion window of the TCP sender at node N0 is recorded in "tcp-example.cwnd".
- "tcp-example.tr" is the ns-3 ASCII trace file containing all simulator events. The ns-3 documentation should explain this trace file in detail. The parts of interest for us are mainly the entries corresponding to a packet getting enqueued and dequeued at the queue of node N0. For example, the enqueue and dequeue entries in the queue of node N0 for packet with sequence number 0 (SYN) at time "1" are as follows.

+ 1 /NodeList/0/DeviceList/0/$ns3::PointToPointNetDevice/TxQueue/Enqueue ns3::PppHeader (Point-to-Point Protocol: IP (0x0021)) ns3::Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 64 id 0 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [SYN] Seq=0 Ack=0 Win=65535)

- 1 /NodeList/0/DeviceList/0/$ns3::PointToPointNetDevice/TxQueue/Dequeue ns3::PppHeader (Point-to-Point Protocol: IP (0x0021)) ns3::Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 64 id 0 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [SYN] Seq=0 Ack=0 Win=65535)

Analyzing simulation results

For your assignment, you will need to run simulations by varying the parameters specified in the table above, and analyze the output files to answer questions about TCP. To understand what is happening with TCP in the simulation, it will be helpful to look at the following metrics.

- **Average TCP throughput at the receiver**. The average throughput can be easily obtained by opening the receiver's pcap file in wireshark, and checking out the "Summary" tab or "Conversations" tab under the "Statistics" menu. (For calculation of average throughput, counting the bytes in TCP headers is optional - either way shouldn't matter much.)
  Alternatively, you can write a script to analyze the pcap file and calculate throughput. You can use "tshark -r" or "tcpdump -r" with several useful options to read the pcap files and process them. Please read up online about the tshark tool and its various options. For example, if you run tshark with the "-T fields" option, you can extract only a subset of fields from the pcap file to the terminal output, which you may then redirect to a smaller

file that is easy to parse. For example, the following command will read the pcap file "tcp-example-0-0.pcap" and extract the timestamp, source IP, destination IP, port numbers, and sequence numbers.

$tshark -r tcp-example-0-0.pcap -T fields -e frame.time_epoch -e ip.src -e ip.dst -e tcp.port -e tcp.seq -e tcp.len -e tcp.ack

You can use several such simple tshark commands to easily extract only the information you care about from the pcap files. After you extract information from the pcap file, you may use sed/awk/perl/python or even C++/Java to extract the various columns in the output, and perform any manipulations (e.g., add up the packet length received column and divide by total time to get throughput).

- **Evolution of the sender's congestion window (cwnd) over time.** The cwnd trace file, generated as part of the simulation output, is written whenever cwnd changes. So simply plotting all the values in the cwnd trace file is enough for you to visualize what happened to the cwnd during the simulation. You may use any graph plotting software. For example, here is a simple gnuplot script, say, "example.gpp", that generates a cwnd vs time plot.

set term postscript eps color
set output 'cwnd.eps'
set ylabel 'cwnd'
set xlabel 'time'
plot 'tcp-example.cwnd' using 1:2

Store the 5 lines above in a file called "example.gp", and store the cwnd trace file generated by ns-3 called "tcp-example.cwnd" also in the same folder. Then run "gnuplot example.gp" at the command line. The graph "cwnd.eps" will be created in the same folder showing the cwnd vs time plot.

In general, once you generate the output data to plot, a simple variant of a gnuplot script like this will let you generate any graph you want.

- **Queue length (occupancy) at the sender's buffer as a function of time.** To understand the behavior of the TCP bottleneck router's buffer, it will be helpful to plot the queue length (occupancy) as a function of time. For the queue length graph, you will have one point on the graph for every change in the queue length, that is, for every time a packet gets enqueued or dequeued. For example, if a packet gets added to the queue at time "t", taking the number of packets in the queue to "n", then you will have a point on the graph corresponding to "t" on the x-axis and "n" on the y-axis. You must analyze the "tcp-example.tr" file generated after the simulation to obtain information on enqueue and dequeue events. You may process this trace file using a simple bash/perl/python script. Once you generate a text file with time and queue length columns, you can easily plot it with gnuplot to visualize how the queue length varied during the simulation.

- **Queueing delay (wait time in queue) at the sender's buffer as a function of time.** Another metric of interest is the amount of time packets spent in the queue (queueing delay or waiting time) as a function of time. For this queueing delay graph, you will have one point on the graph for every queueing delay sample you get, which will be every time a packet is dequeued. That is, if a packet is dequeued at time "t" after waiting for time "w" from the time it was enqueued, then you will have a point on the graph corresponding to "t" on the x-axis and "w" on the y-axis. Much like the queue length, the queue delay can also be obtained from the trace file.

Problem Statement

Q.1. Run the simulation with the default parameters (provided in the table) and answer the following questions.

    a. What is the maximum expected value (theoretical) of throughput (in Mbps)? Why? [1+1]
    b. How much is Bandwidth-Delay-Product (BDP)? Express your answer in terms of the number of packets. [1]
    c. What is the average computed throughput of the TCP transfer? [2]
    d. Is the achieved throughput approximately equal to the maximum expected value? If it is not, explain the reason for the difference. [1+1]
    e. Plot Congestion Window (CWND) with time [1]
    f. Plot queueing delay with time [1]
    g. Are the plots in 1(e) and 1(f) related? [1]

Q.2. Change queue size to 1000 (rest of the parameter values are same as default values)

    a. What is the average computed throughput of the TCP transfer? [1]
    b. Plot CWND with time [1]
    c. Plot queueing delay with time [1]
    d. Compare CWND plots of Q.1. and Q.2.; what insights did you gain? [1]

Q.3. Change N1-N2 bandwidth to 10 Mbps and N1-N2 delay to 100ms (rest of the parameter values are same as default values)

    a. What is the average computed throughput of the TCP transfer? [1]
    b. Plot CWND with time [1]
    c. Plot queueing delay with time [1]
    d. Compare queuing delay plots of Q.1. and Q.3.; what insights did you gain? [1]