**CS 537**

**Project 2, TCP Analysis using network simulator ns2 (100 points)**

This assignment is to be done in two-person groups. Do not show your code to other groups and do not look at other students' code. Prevent other students from accessing your code (do not put it in a public directory). However, you are free to use any tools you like (library, search engines), to discuss problems with others, and to seek the help of the instructor.

#### 1. Study the attached ns2 tutorial

Refer to the provided ns2 tutorial and the given examples to learn about simulations in ns2. Note that the tutorial introduces AWK text-processor to process the default trace file created by ns2, however in this project, we create our own trace file, which contains only the required data. We will not use AWK in this project. Study the tutorial as much as required to learn ns2 simulations.

**2. Objective**

In this project, you will use ns2 simulator for a comparison of congestion control algorithms of different versions of the TCP protocol. The relevant reading in the book is Section 3.7, and you can find additional background on TCP in Section 3.5. Detailed information can be found in RFC793.

**3. Report.docx**

Create an MS Word file named report.docx and answer the yellow highlighted questions given in this assignment and enter your answers into your report file. You may add charts or screenshots as well to support your answers.

**4. Instructions**

1. Download the attached Reno.tcl from Cougar Courses and upload it to your account on cs436.cs.csusm.edu server. You can use WinSCP or FileZilla on Windows or scp or sftp in a terminal on MacOS/Linux. Ns2 has been installed on this server and your account has access to this server. You may also install and use ns2 on your own system. In that case, make sure that you install the last version, which is ns2.35.

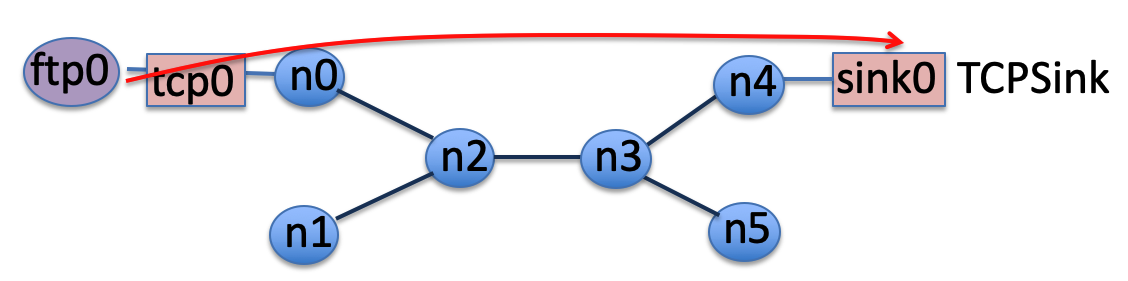
1. If your system is Mac or Linux, you can directly install ns2.
2. If your system is Windows, install a Virtual Machine, e.g. VMware or VirtualBox, and then install ns2 on it.

2. Login to your account on cs436.cs.csusm.edu server. You can use PuTTY on Windows or ssh on MacOS/Linux. Create a folder named Reno and move your file to that folder. You will create several files in this project and creating a separate folder helps to manage the project files.

Run the tcl file with the following command.

ns Reno.tcl

This command will successfully run the tcl file and generate 3 output files: trace.tr, cwnd.tr and thrpt.tr. Download these files to your own system to process and plot charts.



trace.tr: This file contains all the events occurred during the simulation for each packet on each node. This file is automatically created by ns2. Each record of this file follows the format explained in the given ns2 tutorial file.

cwnd.tr: This file contains the size of congestion window at each 0.1 seconds on tcp0 agent. The contents of this file are created by congWin procedure designed by the programmer. This procedure uses cwnd\_ parameter, which is predefined in ns2 to store the congestion window size at a tcp agent.

thrpt.tr: This file contains the amount of ACKed data arrives at sink0 at each 0.5 seconds. Note that sink0 is connected to tcp0 agent. The contents of this file are created by throughput procedure designed by the programmer. This procedure uses bytes\_ parameter, which is predefined in ns2 to store the amount of ACKed data arrive at a sink.

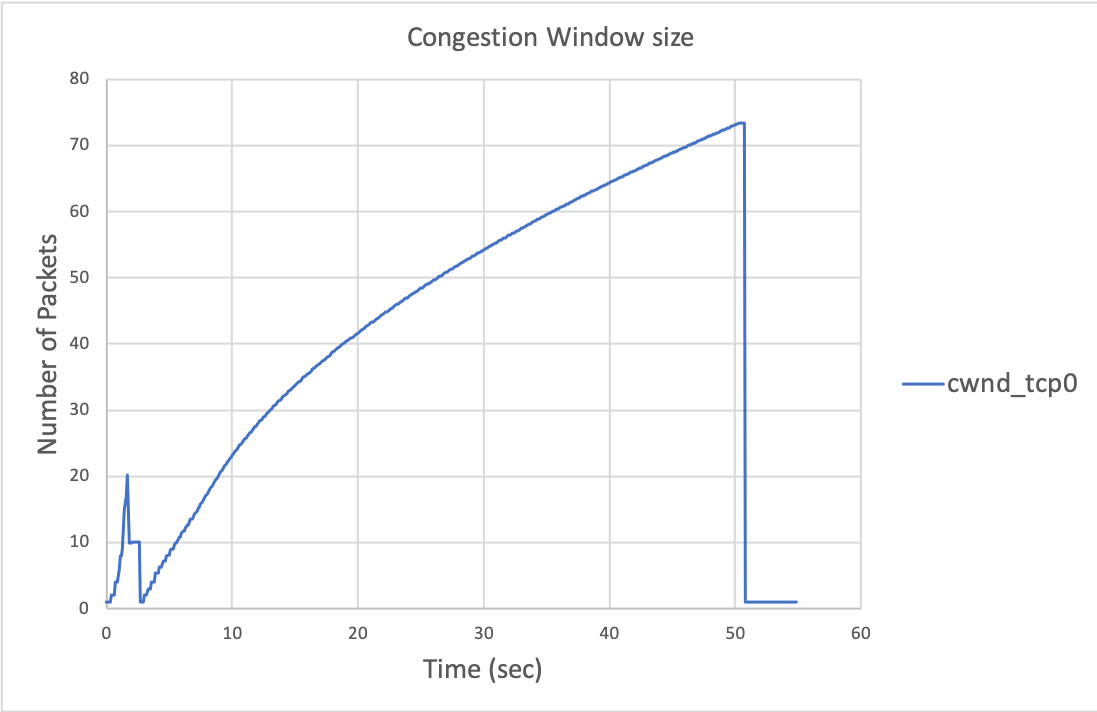
Note that in this procedure, we manually reset the value of bytes\_ parameter at the end of each interval ($sink0 set bytes\_ 0) to separate the amount of throughput at each particular interval. If you remove this command, the procedure will record the total amount of data arrived at sink0 since the beginning of the simulation.

3. Open a new Excel file. You may use any other spreadsheet either, but the instructions given here assume you use MS Excel and store the file in .xlsx extension. Name it Results.xlsx

4. Open cwnd.tr and copy the contents with Ctrl+a and then Ctrl+c and paste them into Results.xlsx. While the first column is still selected, in the “Data” menu, select “Text to Columns”. In the opened window, select “Delimited” and go to the next step. Check both “Tab” and “Space” Delimiters and finish the conversion. It will separate the time and cwnd\_tcp0 columns. Change the sheet name to “cwnd”. Select the two columns and insert a Scatter chart. Enter the proper axis, title on top and legend on the right. Save the chart as a png picture. The following link is a good reference:

<https://support.office.com/en-us/article/save-a-chart-as-a-picture-254bbf9a-1ce1-459f-914a-4902e8ca9217>

The expected result is as following:



5. Open thrpt.tr and copy the contents into a new sheet on Results.xlsx. Split the two columns and plot a chart. Change the sheet name to “throughput”. Enter the proper axis, title on top and legend on the right. Save the chart as a png picture.

6. Open trace.tr file. Refer to the ns2 tutorial file to learn the format of each record in this trace file. Observe that at time 0.1 second, node n0 starts to establish a tcp connection with node n4. The default size of payload for ns2 packets is 1000 bytes. Observe that initially a small packet of size 40 is sent from source. This is a SYN message and 40 is the size of IP header (20) plus the size of TCP header (20). So the payload of this SYN message is actually zero. You observe that at each hop, 3 events occur to the packet, e.g. at node 0, the following events occur.

+ 0.1 0 2 tcp 40 ------- 0 0.0 4.0 0 0

- 0.1 0 2 tcp 40 ------- 0 0.0 4.0 0 0

r 0.120064 0 2 tcp 40 ------- 0 0.0 4.0 0 0

Observe that the source is 0.0 (node n0, port 0 at source) and destination is 4.0 (node n4, port 0 at destination). The packet has entered into the queue at node n0 (+ event at time 0.1 second). The queue is empty, so the packet is immediately loaded into the link to node n2 (– event at time 0.1 second). Then the packet arrives at node n2 (r event at time 0.120064). Calculate the transmission delay and propagation delay and show that the amount of delay matches the time at which the packet arrives at node n2. Note that the packet size is 40 Bytes and it starts to load into the link at 0.1 seconds.

Observe that the packet is sent to the next hop (from n2 to n3) through the same sequence of events (+, –, r).

+ 0.120064 2 3 tcp 40 ------- 0 0.0 4.0 0 0

- 0.120064 2 3 tcp 40 ------- 0 0.0 4.0 0 0

r 0.220704 2 3 tcp 40 ------- 0 0.0 4.0 0 0

Finally, the SYN packet reaches the destination node n4 at 0.240768 seconds, and n4 replies an ACK message. Again 9 events occur back-to-back to the packet so that it reaches node n0.

Now that a TCP connection is established, the tcp agent at node n0 sends packets of 1040 Bytes (1000 Bytes payload + 20 Bytes TCP header + 20 Bytes IP header) to the destination node n4.

7. In trace.tr file, find “d” events. Those are packets that are entered into a queue but then dropped since the queue has had no more slot to handle it. Now compare the moments that a drop occurs in the trace file with the fluctuations in the cwnd and throughput charts. Write down your observations and conclusions in your report file. Note: you may need to enlarge the scale of time axis on the chart so that clearly see at what time the drops on congestion window occur and what is the shape of that. Add pictures of cwnd and throughput charts into your report file and refer to these pictures in your discussion to support your answers.

For example, you may write your discussion as following:

At time XXX a drop occurs. This drop is expected because XXX. As a result of this drop, we observe that cwnd changes XXX and throughput changes XXX. Also other parameters XXX affect these changes in the charts in the following way: XXX.

8. Now create a new folder named Tahoe. Copy your Reno.tcl file to the new folder. Change the file’s name to Tahoe.tcl. Then, make only one change in the new file: change the TCP agent from Reno to the default. To make this change, simply delete “/Reno” on the line you create tcp0. So it will be “set tcp0 [new Agent/TCP]”. Note that Tahoe is the default congestion control algorithm in ns2. Now run the file, create the charts, save them as png pictures. Now for this new scenario, compare the moments that a drop occurs in the trace file with the cwnd and throughput charts. Write down your observations and conclusions in your report file. Also compare the two congestion control algorithms referring to the results.

9. Refer to the following webpages and select two other types of TCP agents implemented in ns2, change your tcp agent to them, run the tcl file, create the charts. Now for these new scenarios, compare the moments that a drop occurs in the trace file with the cwnd and throughput charts. Write down your observations and conclusions in your report file. Also compare all the **four** TCP agents referring to your results.

<https://www.isi.edu/nsnam/ns/doc/node387.html>

<http://netlab.caltech.edu/projects/ns2tcplinux/ns2linux/tutorial/index.html>

Note: TCP SACK1 and TCP Linux are commonly used. You may choose these two. Note that some algorithms need some initial settings to show their effectiveness. For example, TCP Linux needs a large initial cwnd\_.

**Important note:**

Study the structure, design and implementation of the two TCP agents that you chose. You should show the difference between those two TCP agents and Tahoe and Reno in your charts and results and also justify the differences referring to the designs and implementations of these 4 different algorithms. Find scenarios that result in a significant difference between the performances of Reno, Tahoe, and the other two TCP agents.

**Step2:**

1. Modify the script Reno.tcl to simulate the following configuration.

# Create the links:

$ns duplex-link $n0 $n2 2Mb 100ms DropTail

$ns duplex-link $n1 $n2 2Mb 100ms DropTail

$ns duplex-link $n2 $n3 0.2Mb 500ms DropTail

$ns duplex-link $n3 $n4 2Mb 100ms DropTail

$ns duplex-link $n3 $n5 2Mb 100ms DropTail

# Create a bottleneck with a maximum queue size of 5 packets

$ns queue-limit $n2 $n3 6

Create a UDP agent and attach it to node n1. Then create a cbr traffic generator and attach it to the UDP agent. Both of the TCP and UDP sources should start transmitting at time 0.1 second and stop at time 50.0 seconds. Configure the UDP source to be a CBR traffic generator, sending packets of size 1000 bytes, at a rate of 0.1Mbps. Configure the UDP sink to be a LossMonitor agent. Keep the same TCP source and sink as in the script Reno.tcl. To do this, add the following script to your tcl file.

#Establish a UDP connection between n1 and n5

set udp1 [new Agent/UDP]

$ns attach-agent $n1 $udp1

set udpsink1 [new Agent/LossMonitor]

$ns attach-agent $n5 $udpsink1

$ns connect $udp1 $udpsink1

# Create a CBR traffic generator at source n1 and attach it to udp1

set cbr1 [new Application/Traffic/CBR]

$cbr1 set type\_ CBR

$cbr1 set packet\_size\_ 1000

$cbr1 set rate\_ 0.05Mb

$cbr1 set random\_ false

$cbr1 attach-agent $udp1

$ns at 0.1 "$cbr1 start"

$ns at 50.0 "$cbr1 stop"

You also need to add some scripts to the throughput procedure to record throughput at udpsink1. The procedure will change as following:

proc throughput {} {

global ns f1 sink0 udpsink1

set now [$ns now]

set interval 0.5

set acked0 [$sink0 set bytes\_]

set rcved1 [$udpsink1 set bytes\_]

#Calculate the throughput (in Mbit/s)

#Throughput is the amount of ACKed data reached the destination

set thr0 [expr $acked0/$interval\*8/1000000]

#Throughput is the amount of UDP data reached the destination

set thr1 [expr $rcved1/$interval\*8/1000000]

###Print TIME THROUGHTPUT into the output file

puts $f1 "$now $thr0 $thr1"

#Reset the bytes\_ values on the traffic sinks

$sink0 set bytes\_ 0

$udpsink1 set bytes\_ 0

$ns at [expr $now+$interval] "throughput"

}

#Print Column names at the first row of output file

#This line will be used to plot the chart

puts $f1 "Time Throughput\_tcpsink0 Throughput\_udpsink1"

2. Modify the script and repeat the simulation so that the UDP source sends at a rate of 0.1 Mbps.

3. For each of these two simulations, run the tcl file and plot the charts. Save the charts as png pictures. Now compare the moments that a drop occurs in the trace file with the cwnd and throughput charts. Write down your observations and conclusions in your report file. Explain how TCP and UDP interact in each scenario when udp rate increases. Add the plots pictures and refer to them in your report file to support your discussion.

**Step3:**

1. Write a script that simulates the following topology. Note that you should create nodes n0-n9 before connecting them through links.

# Create the links:

$ns duplex-link $n0 $n4 2Mb 100ms DropTail

$ns duplex-link $n1 $n4 2Mb 100ms DropTail

$ns duplex-link $n2 $n4 2Mb 100ms DropTail

$ns duplex-link $n3 $n4 2Mb 100ms DropTail

$ns duplex-link $n4 $n5 0.2Mb 500ms DropTail

$ns duplex-link $n5 $n6 2Mb 100ms DropTail

$ns duplex-link $n5 $n7 2Mb 100ms DropTail

$ns duplex-link $n5 $n8 2Mb 100ms DropTail

$ns duplex-link $n5 $n9 2Mb 100ms DropTail

# Create a bottleneck with a maximum queue size of 5 packets

$ns queue-limit $n4 $n5 6

Attach a TCP agent to each of nodes n0-n3. Attach one TCPSink agent to each of nodes n6-n9. Create one FTP traffic generator for each tcp source and connect the tcp agent to the corresponding tcp sink, i.e. connect tcp0 at node n0 to sink0 at node n6, etc.

The ftp sources start at different times: ftp0 starts at 0 second, ftp1 starts at 5 seconds, ftp2 starts at 10 seconds, ftp3 starts at 15 seconds. All sources stop transmitting at time 100 seconds. Stop your simulation at 105 seconds.

2. Modify the throughput and cwnd procedures to record the information of four TCP flows into one tr file for throughput and one tr file for cwnd. Plot the two charts and save them as png files. Add the pictures into your report file.

3. At what time all flows reach a fair share of the available bandwidth? What is the average throughput for each flow during the time where all flows get a fair share of bandwidth? Calculate the averages in your Excel file. Provide a **discussion** of the fairness referring to the results you got from throughput and cwnd charts and also drop events in your trace.tr file. You may need to add more charts that enlarge the scale in some part of a chart or separate some flow from other flows to clarify your discussion. Refer to the book, section 3.7.1.

**5. Submission**

Create a folder containing all tcl and Excel files including charts that you created at each step. Also add the most important part of your submission, the report file including your answers to the given questions and analysis. Compress the folder in a zip file and submit that to your assignment.