Lab 04: NS2 based study of the performance of CSMA/CD (802.3)

OSL, Fri Feb 12, 2015

Objective:

Understand the performance of CSMA/CD (MAC protocol employed by Ethernet) in terms of system efficiency and fairness measure.

General instructions:

1. This lab is to be done in **groups of two students**
2. Download the file “lab04-student-files.tgz” and unzip it. You will find all relevant files needed for this lab in this directory.
3. Create a directory called <rollnumber1>_<rollnumber2>_lab04. As you proceed with the lab instructions below, note down observations or relevant output from whatever you do in a file named ”lab04.txt” using a text editor.
4. Also add to this directory any written code along with output files. You will find more details of this in the specific exercises.

Lab Instructions:

Exercise 1: CSMA Warmup [16 Marks]

NS2 simulator architecture does not quite capture reality and sometimes it does retro-fitting to get the intended result. So, be prepared to be confused and amused by this. But hopefully by the end of this warmup, you will learn enough to proceed with actual experimentation of the MAC protocol.

Step1: Understanding the script

For this exercise, you will be using the file ns-csmacd-lab04.tcl to run the experiments. This script essentially creates a LAN based on 802.3(CSMA/CD), which is the name of the MAC standard for Ethernet. It takes as inputs the number of nodes that will be part of the LAN and a seed for the random number generator. Carefully go through the script file and answer the following in lab04.txt.

1. Which command was used to create the LAN? What are its arguments? How was the MAC protocol specified?
2. What is the data rate (bandwidth) of the LAN? How did you determine it?
3. At what rate (kbps) is each node in the LAN generating traffic? How did you determine it?
4. What is the packet size (bytes) used in the generated traffic? What is the interpacket gap? How did you determine it?
5. What is the duration of the simulation when **traffic is active**? How did you determine it?
6. Which node is the destination for all the traffic? How did you determine it?

Step2: Executing the script

Now that you understoof the script, execute it with number of nodes as 4 and seed as “1234”. Basically: “ns ns2-csmacd-lab04.tcl -nn 4 -seed 1234”.

Look at the NAM output. Magic! Its buy 4, get 2 free! And one hidden gift too!

- You will see a total of 6 nodes in NAM, even though your argument specifies 4. Node 0 in this exercise is a sink node i.e it does not generate any traffic (and hence does not contend for the channel) and just sinks traffic generated by other nodes.

- Nodes of relevance to us are nodes 1,2,3,4. These are the stations you specified via “-nn” argument. These are the ones that are contending for the channel.
- Node 5 is not listed in NAM but if you look at the output file “csmacd.tr”, you will see node 5 listed. The LAN bus is modelled in ns2 as a node instead of as a link, where any node wishing to transmit a packet on the link, essentially gives it to node 5. You will see this operation listed by a “h” in the output file.
- Node 6 is again a dummy node, created just to get the orientation right for easy visualization in NAM. You can ignore this node and the corresponding link between it and node 0. (For added fun, you can comment the code corresponding to node6, to see how Nam orients the nodes without it.)

Step3: Analysing the results

Let us process the results for one sample run before we perform other runs. Look at the csmacd.tr file generated above using nodes as 4 and seed as 1234. Dig in a bit to figure out what is happening. 'c' in the trace stands for collisions. You can also run the code with nodes as 1 to contrast the results. But to answer below questions, run using nodes as 4 and seed as 1234.

Answer the following in lab04.txt. Also specify how you determined the answer.

1. What are the total number of packets (P) succesfully received by the reciever?
2. Are there any packets dropped (at different senders) during the run? Why or why not?
3. What is the system throughput and efficiency? System throughput is defined as the total number of bits succesfully received (P * packet size * 8) divided by the simulation duration. System efficiency is the ratio of system throughput by the data rate of the LAN. Express system efficiency as a percentage.
4. Why is the system efficiency not 100%?
5. What are the individual throughputs of nodes 1,2,3 and 4? For this, you need to calculate the number of packet received succesfully at node 0 from nodes 1,2,3 and 4 respectively. So, a total of 4 values, one corresponding to each node. Check that the sum of individual throughputs sums to the system throughput calculated in Q.3
6. Trace the collision profile for the first packet (sequence number 0) generated by nodes 1,2,3 and 4? In other words, what is the time at which these packets got generated? When did they get transmitted? Did they collide? When were they received. Who won the collision in which order etc.... You just need to look till time 0.54 sec in the csmacd.tr to figure this out.

Step 4: Digestion

Goal: When evaluating MAC protocols, some commonly used metrics to evaluate their effectiveness are system efficiency, throughput, packet loss, delay, fairness measure.

- System efficiency captures how good the MAC is in making efficient use of the link capacity (1Mbps in the example). In MAC protocol design, we want this number to be as close to 100% as possible under high load. Any link capacity wastage in terms of collisions, keeping link idle (during backoffs) reduces efficiency.
- At the individual node level, one likes to measure what throughput, packet loss and delay the packets of the flow are achieveing since these have direct bearing on end user experience.

- Fairness is another very useful metrics when dealing with MAC protocols. An ideal MAC will share the resources among all users in a fair manner (equally is all carry equal weight). Readup on Jain's fairness index (http://en.wikipedia.org/wiki/Fairness_measure), which is commonly used in evaluations.

Take some time to digest what the script does and how you can use the script to evaluate the performance of MAC protocols in light of above metrics. Basically what experiments would you design, how would you evaluate the trace files, what would you plot etc.

In the interest of time, I will provide detailed instructions on evaluation methodology in next few exercises, but before you see them, do think on your own.

Exercise 2: Throughput, Efficiency and Packet Drops [16 Marks]

When dealing with MAC protocols, one can get good insight into its efficiency by varying the number of nodes in the system. In this exercise, you will vary the number of nodes. For this, you need to automate this process using a 'for' loop in bash. **Important: For these runs, specify the seed as "sum of the last four digits of you and your partner's roll numbers". Also comment out the line "exec nam out.nam", otherwise you will have to close a lot of nam windows.**

1. Vary the number of nodes from 1 to 20 i.e. nn takes on values 1,2 ....., 20 and for each run specify the seed as calculated above based on your roll numbers. This gives a total of 20 runs.
2. For each run, calculate the system efficiency, system throughput and percentage drops (total dropped packets/total generated packets). Output this to a file overall-stats.txt, where the first column is the number of nodes, the second column is the system throughput, the third column is the system efficiency and the fourth column is the drop percentage. In summary after all the runs, the file should have 20 rows and 4 columns, one row generated from each run.
3. Use gnuplot to generate three graphs. One plots system throughput as a function of the number of nodes ("my-thr.eps"), the second plots the system efficiency (my-eff.eps") as a function of nodes. The third plots drop-percentage as a function of nodes (my-drops.eps). Look at the sample "thr-nodes.eps" for an idea on how the plot looks.
4. Comment on the graphs plotted in lab04.txt i.e. explain why the graphs are the way they are, any interesting observations, what may have contributed to the observed numbers etc. Specifically comment on when nodes are 8 (offered load < 1Mbps) vs nodes are 12 (offered load > 1Mbps).
5. Given the link capacity of 1Mbps and by looking at the graphs, what upper limit would you impose on the number of nodes in the LAN if you were the system administrator?
6. Include your bash scripts, overall-stats.txt, my-thr.eps, my-eff.eps, my-drops.eps in the submission directory. **DO NOT** submit the trace files for these runs since they will be quite a few.

Exercise 3: Fairness Run [8 Marks]

Run the experiment by setting the number of nodes as 20 and the seed corresponding to your roll numbers. For this run, calculate the individual throughputs for each of the individual nodes i.e. 1,2,3......20. Output this to a file "run20-node-stats.txt", where the first column is the node id and the second column is the throughput obtained by that individual node. This file will have 20 rows and 2 columns. Use gnuplot to plot the individual node throughputs ("my-fairness.eps"). Look at the sample "fairness.eps".

1. Comment on the graphs obtained i.e. explain why the graphs are the way they are, any interesting observations, what may have contributed to the observed numbers etc.
2. Use the Jain's fairness index (http://en.wikipedia.org/wiki/Fairness_measure) to comment on the fairness of Ethernet.
3. Submit the trace file csmacd.tr, run20-node-stats.txt and my-fairness.eps

Exercise 4: Delay [Extra Credit: 10 Marks]

Run the experiment by setting the number of nodes as 20 and the seed corresponding to your roll numbers. This is the same run as the fairness run, so you don't need to run it again, just work with the trace. For every packet generated in the run, calculate two time values

- Time from enqueue (+) to dequeue (-) (termed EnqToDeq): This time captures the time spent in queue/buffer at the interface
- Time from Deque (-) to receive (r) (termed DeqToRecv): This time captures the time spent in accessing the channel (collisions and final succesful transmission).

Output these values to a file “run20-delay-stats.txt”, where the first column is the packet no, the second column is “EnqToDeq” delay and third column “DeqToRecv” delay. Plot (as a scatter plot) EnqToDeq delays (EnqToDeq.eps) and also DeqToRecv delays (DeqToRecv.eps).

1. Comment on the graphs obtained i.e. explain why the graphs are the way they are, any interesting observations, what may have contributed to the observed numbers etc.
2. Submit file “run20-delay-stats.txt”, EnqToDeq.eps and DeqToRecv.eps

Submission instructions

The directory named <rollnumber1>_<rollnumber2>_lab04 that you will submit should contain the following files:

1. lab04.txt
2. bash script
3. overall-stats.txt
4. my-thr.eps
5. my-eff.eps
6. my-drops.eps
7. csmacd.tr (corresponding to the run with 20 nodes and roll number seed)
8. run20-node-stats.txt
9. my-fairness.eps
10. run20-delay-stats.txt (optional)
11. EnqToDeq.eps (optional)
12. DeqToRecv.eps (optional)

Now tar it as follows:

```
tar -zcvf <rollnumber1>_<rollnumber2>_lab04.tgz <rollnumber1>_<rollnumber2>_lab04/
```

Submit the file <rollnumber1>_<rollnumber2>_lab04.tgz via moodle for grading.