# CS3210 Computer Networks Lab
# Even Sem. 2017, Prof. Krishna Sivalingam
# Lab 9: TCP Congestion Control
# Due date: April 8, 2017, 11PM, On Moodle
# Individual Assignment

March 29, 2017

## 1 Description

*Note: Viva voce exam and demo might be conducted in the presence of the instructor.*

The objective of this project is to emulate the TCP congestion control algorithm, as explained below.

### 1.1 Assumptions and Variables

The assumptions and variables are given below:

- Receiver Window Size is set to 1 MB, and does not change during the entire duration of the emulation.
- The Sender always has data to send to the receiver.
- Sender's MSS is 1 KB.Each segment has a fixed length of one MSS.
- Go-back-N is used, but cumulative acknowledgments are not considered. For each segment, an individual timeout timer and ACK are used.
- The congestion window is always intrepreted as a multiple of MSS (1 KB).
- The congestion threshold is always set to 50% of the current CW value.
- $K_i, 1 \leq K_i \leq 4$ denotes the initial congestion window (CW). Default value is 1. The initial CW is given by:

$$\mathrm{CW}_{new} = K_i * \mathrm{MSS}$$

- $K_m, 0.5 \leq K_m \leq 2$ denotes the multiplier of Congestion Window, during exponential growth phase. Default value is 1. When a segment's ACK is successfully received,

$$\mathrm{CW}_{new} = \min(\mathrm{CW}_{old} + K_m * \mathrm{MSS}, \mathrm{RWS})$$

- $K_n, 0.5 \leq K_n \leq 2$ denotes the multiplier of Congestion Window, during linear growth phase. Default value is 1. When a segment's ACK is successfully received,

$$\mathrm{CW}_{new} = \min(\mathrm{CW}_{old} + K_l * \mathrm{MSS} * \frac{\mathrm{MSS}}{\mathrm{CW}_{old}}, \mathrm{RWS})$$

- $K_f, 0.1 \leq K_f \leq 0.5$ denotes the multiplier when a timeout occurs:

$$\text{CW}_{new} = \max(1, K_f * \text{CW}_{old})$$

- $P_s, 0 < P_s < 1$, denotes the probability of receiving the ACK packet for a given segment before its timeout occurs.

### 1.2 Running the program

The program is invoked with the following command-line parameters:

```
% ./cw -i <double> -m <double> -n <double> -f <double> -s <double> -T <int> -o outfile
```

The values correspond to $K_i, K_m, K_n, K_f, P_s$ and the total number of segments to be sent before the emulation stops. The output (specified below) is saved in an output file.

The congestion window progression is done on a slot by-by-basis. In each "round" as explained in the class, a set of segments are sent, in proportion to the current value of CW, i.e. $N = \left\lceil \frac{CW}{MSS} \right\rceil$. For example, if CW is equal to 4.3 KB, five packets are sent. However, the CW growth is based on MSS values as explained earlier.

For each segment transmitted, the ACK for this segment is received before timeout with random probability $P_s$, and timeout occurs with probability $(1 - P_s)$. Depending on this outcome, the CW increases and decreases as described earlier.

## 2 What to do?

Given the set of input parameters, the emulation progresses as above. The congestion window value is printed to the output file (one per line) at each CW update. A graph with x-axis being the update number and y-axis the corresponding CW value must be plotted.

A technical report must be written based on the results and graphs obtained for the following parameter combinations:

$K_i \in \{1, 4\}$; $K_m \in \{1, 1.5\}$; $K_n \in \{0.5, 1\}$; $K_n \in \{0.1, 0.3\}$; $P_s \in \{0.01, 0.0001\}$.

The report should explain how these factors influence the CW change over the duration of the session.

## 3 What to Submit?

The platform for this project will be Linux and C/C++/Java. Create a tar-gz file with name: Lab9-RollNo1.tgz (e.g., Lab9-CS14B110.tgz).

The directory should contain the following files:

- Source File(s)
- Makefile and Script File
  Typing command 'make' or your script program, at the UNIX command prompt, should generate all the required executables.
- A Script file obtained by running UNIX command *script* which will record the way you have finally tested your program.
- Technical Report (as above)
- a README file containing instructions to compile, run and test your program.

# 4 Help

1. Ask questions EARLY and start your work NOW (really, no choice). Take advantage of the help of the TAs and the instructor.
2. Submissions PAST the extended deadline SHOULD NOT be mailed to the TAs. Only submissions approved by the instructor or uploaded to Moodle within the deadline will be graded.
3. Demonstration of code execution to the TAs MUST be done using the student's code uploaded on Moodle.
4. NO sharing of code between students, submission of downloaded code (from the Internet, Campus LAN, or anywhere else) is allowed. The first instance of code copying will result in ZERO marks for the Lab component of the Course Grade. The second instance of code copying will result in a 'U' Course Grade. Students may also be reported to the Campus Disciplinary Committee, which can impose additional penalties.
5. Please protect your Moodle account password. Do not share it with ANYONE. Do not share your academic disk drive space on the Campus LAN.

# 5 Grading

- Implementation of CW Scheme: 70 points
- Report: 25 points
- Viva Voce: 5 points