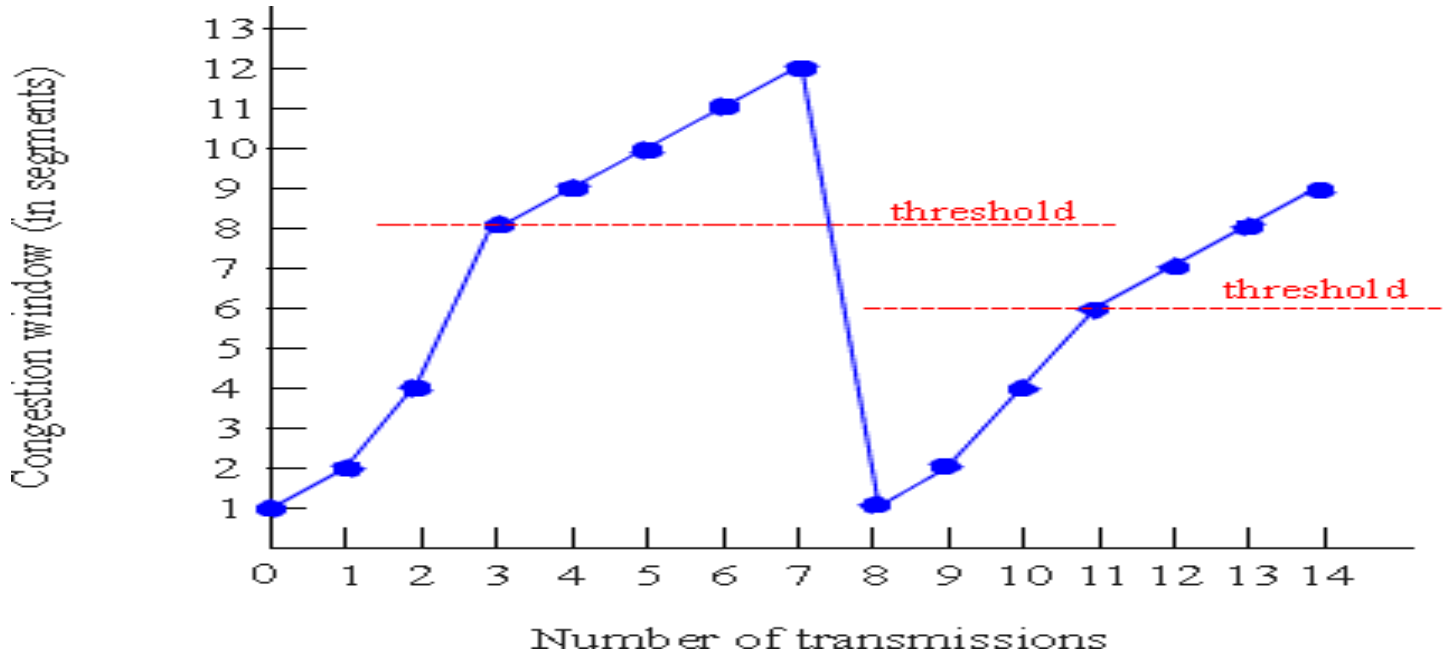
Assignment 5 –
TCP Congestion Control Simulation

Overview



The goal of this experiment is to dynamically vary the sender's congestion window size as per the congestion control algorithm and then plot the change of CW with respect to number of updates made To CW.

What is given

- Receiver Window Size (RWS) = 1 MB
- MSS (Maximum Segment Size) = 1 KB

- Congestion Threshold ~ Slow start threshold (ssthres) = 50% of Congestion window size (Initially, the ssthres value can be set to a reasonable value based on your choice)

- Individual ACKs for every segment sent

What should be the input

- You need to write your program in one file, and name the file as per the guideline mentioned in the assignment document (Download the latest document from moodle)

- The inputs to the program will be:
  - Ki – constant to be used initially at the start of experiment while setting congestion window size
  - Km – constant to be used in exponential growth phase (slow start)
  - Kn – constant to be used in additive (linear) growth phase
  - Kf – constant to be used in multiplicate decrease phase (when congestion occurs)
  - Ps – Probability of receiving an acknowledgement corresponding to a segment
  - T – total number of segments to be sent

What to do

- In this assignment, the sender's congestion window size is dynamically varied using the TCP congestion control algorithm as discussed in the class

- The congestion window will be set to a initial value using the equation:

CWnew = Ki ∗ MSS // CW stands for Congestion Window

- ***Example: If Ki = 2, and MSS = 1 KB, then:***

  - ***CWnew = 2*1 = 2KB,***
  - ***Or, CWnew = 2 segments***

What to do

- Now, once the congestion window is set, and the number of segments to be sent is determined, we need to determine whether segment is lost or not.

- We do this by determining the probability of receiving an ACK for every segment sent – (check next slides).

- If the ACK is received successfully, then no sign of congestion and thus you go for increasing the CW value as shown in the next slides

How to increase the CW size

- Once the initial CW is set, the CW value will be increased first exponentially and then linearly till congestion occurs.

- ***CWnew = min(CWold + Km * MSS, RWS) //*** ***CW size during exponential growth***

- ***CWnew = min(CWold + Kn * MSS/CWold, RWS) //*** ***during linear growth***

- Continue increasing the CW size till you get congestion.

How to decide if Congestion occurs

- The idea here is to simulate the process of sending transport layer segments and receiving ACKs corresponding to each and every segment sent

- For each segment sent, we toss a coin to decide if the sender will receive an ACK corresponding to the segment or not.

(Programmatically, we call a built-in function in C (such as rand()) which will generate random numbers from a uniform distribution and compare the generated random number with the user supplied probability Ps. If the generated random number is less than Ps, it means ACK is lost implying packet loss. However, if generated random number is more than Ps, it means ACK successfully reaches sender.)

- If congestion occurs (i.e., ACK does not reach sender), decrease CW multiplicatively as follows:

- ***CWnew = max(1, Kf * CWold) //*** ***during failure of receipt of ACK***

What is output

- Whenever, you update the CW value (either decrease the CW size during congestion or increase during linear or exponential growth), you print the output in a file. This file write operation should happen for each update to CW
- Output file will consist of the following:

Update Number	CW value
1	1
2	2
3	4
4	8
5	9
6	10

What to do next

- You have to do this experiment for different values of:
  - Ki,
  - Km.
  - Kn,
  - Kf,
  - Ps

- At any point of time in your simulation, you vary any one of the parameters mentioned above, while keeping the other 3 fixed to some value. (See next slide for example)

What to do Next

- **Example**:
- Suppose you want to **study** the behavior of congestion avoidance algorithm with respect to different values of Ki, where Ki varies from 1 to 4.

- So, ***first*** you fix the values of Km, Kn, Kf, and Ps to 0.5, 0.5, 0.3, and 0.5 respectively (These values are just for example, you can take any values of your own).
- ***Second***, you fix the value of **Ki to 1**, and call your program with all the values you just fixed. Every time you run a program, you will get an output file with different values of CW size and update numbers.
- ***Third***, you change the value of Ki to 2, and repeat the above steps.
- ***Fourth***, you continue doing step 2 for Ki = 3 and 4 respectively.

What to do next

- You now get 4 output files corresponding to different values of Ki (where Ki varies from 1 to 4) and Km, Kn, Kf, and Ps are fixed at 0.5, 0.5, 0.3, and 0.5 respectively.

- ***Fifth step,*** Plot a graph by taking input as the 4 files (means you will get 4 curves in a single graph), and understand the behavior of congestion avoidance algorithm with different values of Ki
- ***Sixth Step***, explain your understanding in a report

- **Step seven,** Continue this process for all the different parameters such as Km, Kn, Kf, and Ps. At the end you should be getting 5 graphs. All the graphs as well as your understanding pertaining to the influence of each of these parameters on CW size should be clearly explained in the report.