Faculty of Arts and Science
University of Toronto
CSC 358 - Introduction to Computer Networks, Winter 2019

Solutions for Assignment 5

Question 1

Suppose packets can get dropped or arbitrarily delayed inside a packet network. Suppose that two peer processes are exchanging packets in a session and want to terminate the session. We would like a protocol that exchanges packets in such a way that both peer processes know that they can terminate with the knowledge that no further packets will arrive from the other peer process. Can such a protocol be designed (answer "yes" or "no")?

- **NO** - this problem is equivalent to solving the three-army problem that we discussed in class.

Question 2

In class we discussed that in order to safely establish a TCP connection and avoid packets from different sessions being confused, we need the following mechanisms:

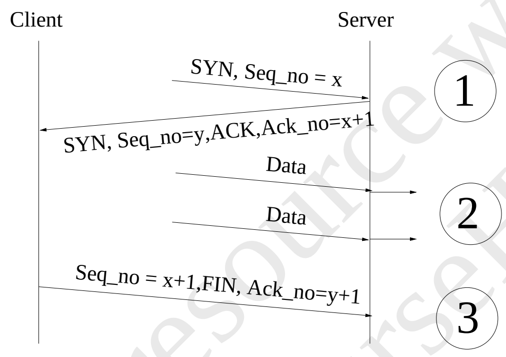
(1) Time-stamps to ensure that packets "die" after some fixed time T.

(2) Clock to determine sequence numbers.

(3) Connection setup through "three way handshake"

Show by means of a counter example that we cannot safely establish a TCP connection without using a three way handshake.

When we only use a two-way handshake, then a delayed (outdated) connection request, and data, from an old connection that has been closed can lead to the situation where the transport layer releases wrong data to the application layer (see figure below).



Two-Way Handshake:

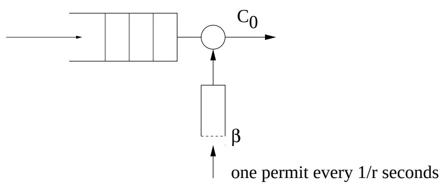
(1) An out-dated (duplicated and delayed) connection request arrives at B. B believes that this a new and valid connection request and opens a new connection.

(2) Data packets from the out-dated connection arrive at B. B believes that this are valid data packets and releases them to the next higher layer.

(3) A realizes that B opened a connection that is no longer active, and sends a connection-termination packet. However, by the time this packet arrives, B has already released outdated data to the next higher layer.

Note that with a three-way handshake, this could not happen. B would wait until it receives a connection-granted, or connection-termination, packet from A. In the above case, B would wait with releasing the data packets until the connection-termination packet arrives (at point (3)). At this point, B realizes that this is not a valid connection and would not release the data packets to the next higher layer.

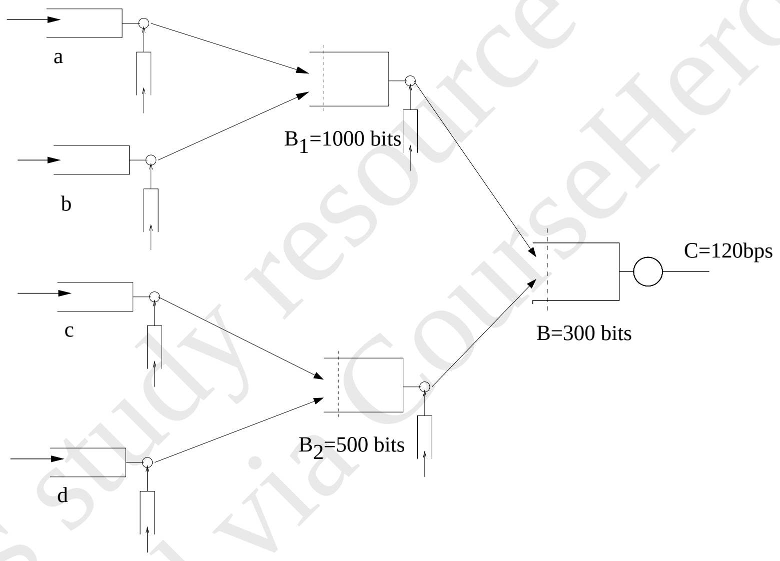
Question 3
Consider the leaky bucket mechanism where (up to β) unused permits can be stored and used later by the session. Assume that all packets have the same length $L = 100$ bits. To



model the leaky bucket, we use a fluid-flow model (kitchen-sink model) and assume that the transmission capacity C_0 of the link after the "leaky-bucket-controlled" buffer is equal to ∞ (very fast). The total amount of data (fluid) $A(t, \tau)$ that a session under the above leaky bucket can submit in the interval $[t, \tau]$ is then upper-bounded by

$$A(t, \tau) \leq Lr(t - \tau) + L\beta.$$

Consider the situation below. Four sessions (a, b, c and d) access a single link with



transmission capacity $C = 120$ bits per second and space (in queue or in service) for $B = 300$ bits. However, before accessing the link, the session go through an additional buffer which is again controlled by a leaky bucket: session a and b go through buffer 1, and session c and d go through buffer 2. Buffer 1 has space (in queue or in service) for $B_1 = 1000$ bits, and buffer 2 has space (in queue or in service) for $B_2 = 500$ bits. The two buffers can again transmit data at a speed of $C_0 = \infty$. The parameters for the system are as follows,

$$r_a = 0.2 \text{ permits per second}, \qquad \beta_a = 4,$$

$$r_d = 0.1 \text{ permits per second}, \qquad \beta_d = 2,$$

$$r_1 = 0.5 \text{ permits per second}, \qquad \beta_1 = 1,$$

What are the maximal rates r_b, r_c, and r_2, and the maximal number of saved permits β_b, β_c, and β_2, that we can allocate to the corresponding leaky buckets so that we can guarantee that there will never be a buffer overflow (packet loss) at either of the 3 buffers with space B, B_1, and B_2, respectively?

We have the following conditions.

- For buffer (C, B)

$$L(r_1 + r_2) \le C$$
$$L(\beta_1 + \beta_2) \le B$$

- For buffer (C_0, B_1)

$$L(r_a + r_b) \le Lr_1$$
$$L(\beta_a + \beta_b) \le B_1$$

- For buffer (C_0, B_2)

$$L(r_c + r_d) \le Lr_2$$
$$L(\beta_d + \beta_c) \le B_2$$

It follows that

$$r_2 = 0.7$$
$$\beta_2 = 2$$
$$r_b = 0.3$$
$$\beta_b = 6$$
$$r_c = 0.6$$
$$\beta_c = 3$$

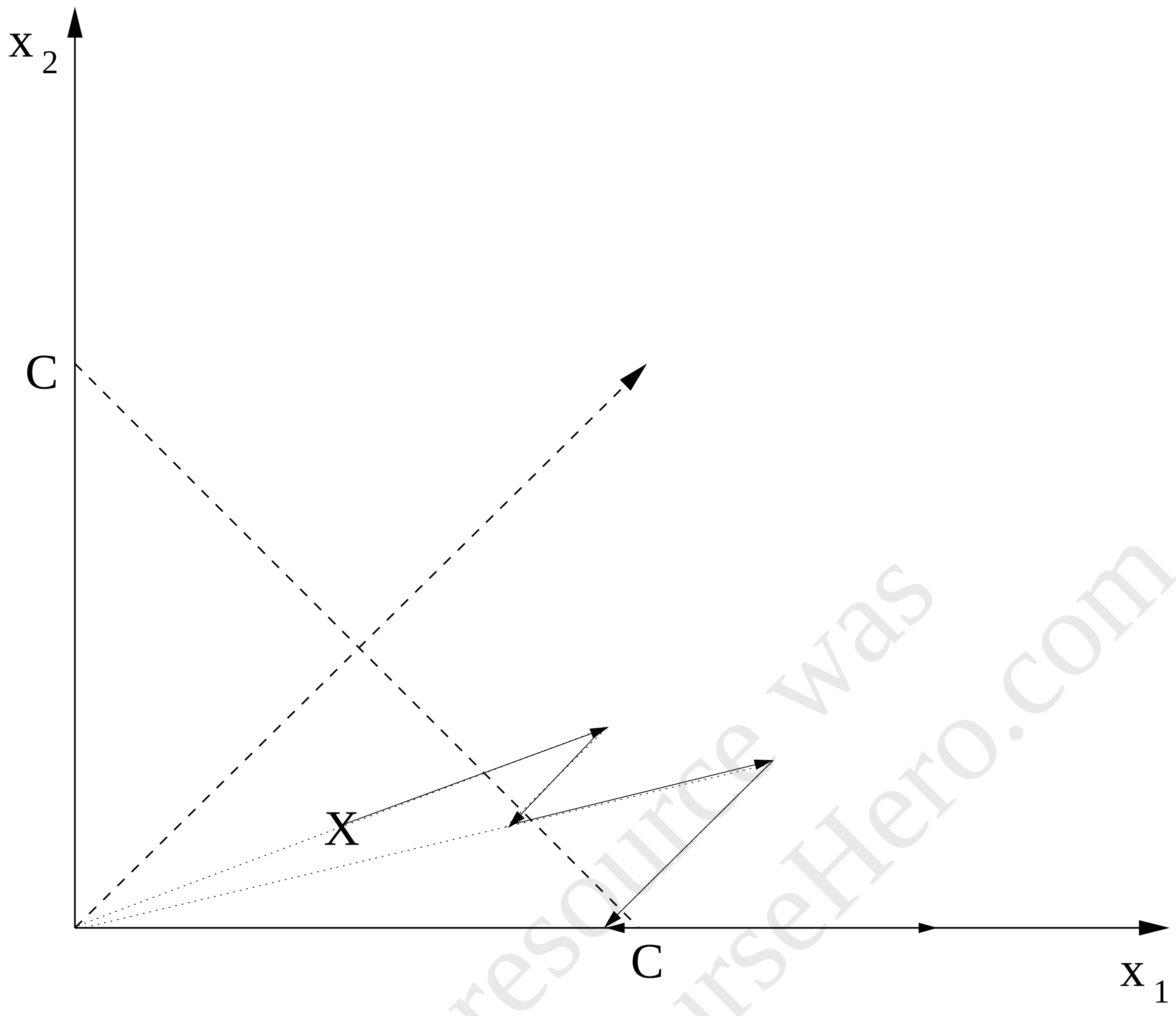
Question 4

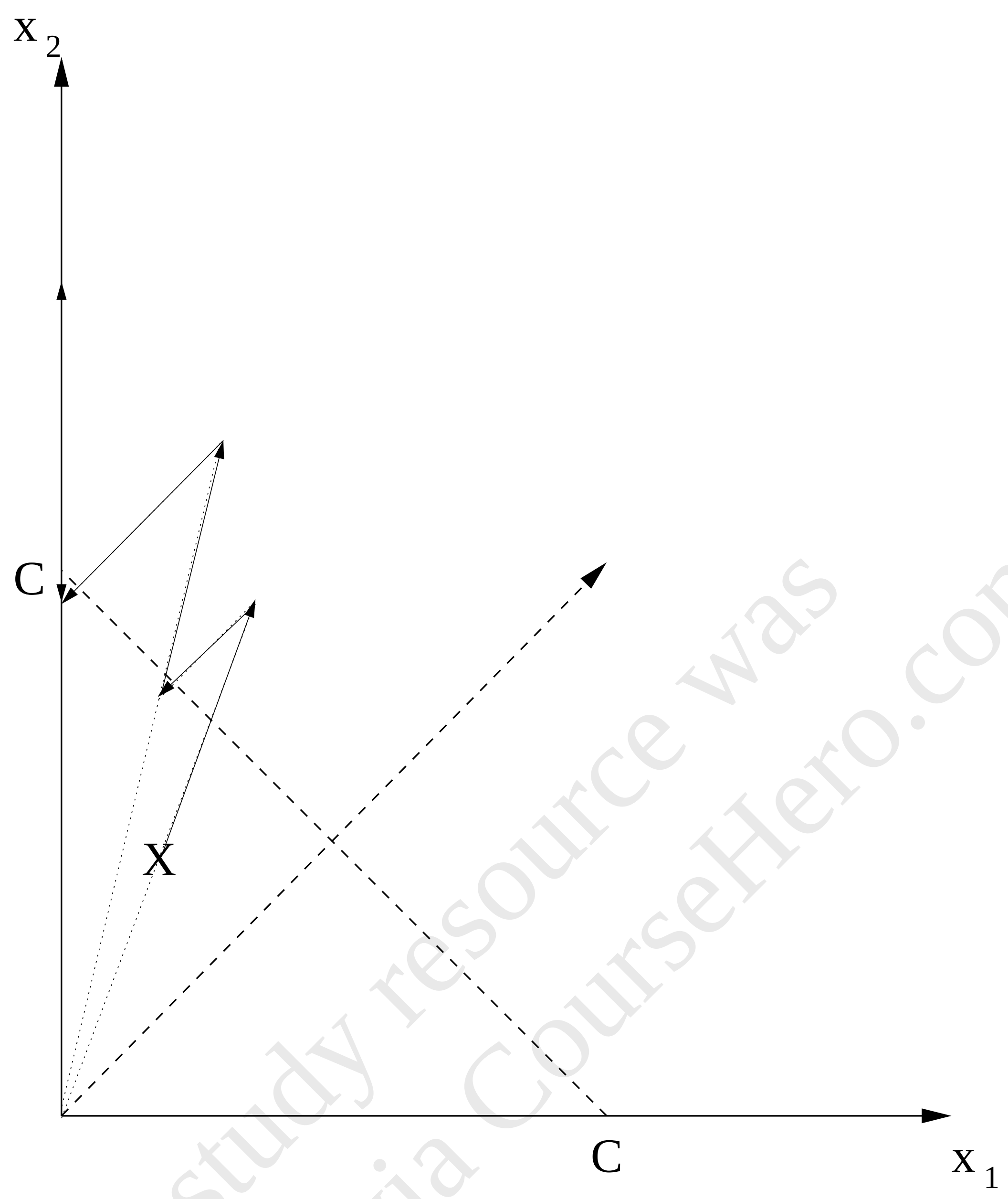
Consider two sessions, 1 and 2, which share a single link with capacity C. Assume that both sessions use a congestion window (as in TCP) to control the transmission rate. Let w_1, and w_2, be the window size of session 1, and 2, respectively; and let x_1, and x_2, be the corresponding transmission rate of sessions 1, and 2, respectively.
For this situation, we used in class a simplified model to analyze the AIMD (additive-increase, multiplicative-decrease) algorithm of TCP congestion control, where we ignored the slow start phase and assumed that the threshold value $w/2$ is constant (steady-state). Using the same model, we can analyze the MIAD (multiplicative-increase, additive-decrease) algorithm for adjusting the congestion window size w_1 and w_2, where we set for $n = 1, 2$

$$w_n = a_I w_n, \qquad \text{when } (x_1 + x_2) < C$$

$$w_n = w_n - 1, \quad \text{when } (x_1 + x_2) \geq C$$

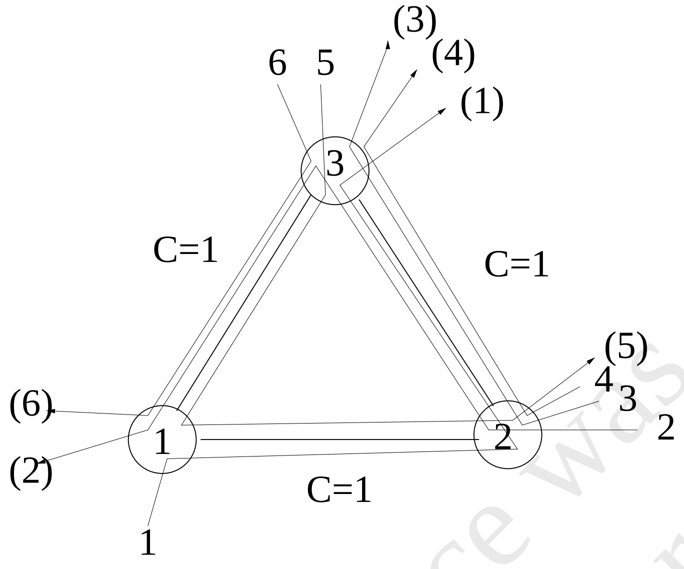
where $a_I > 1$. Show the dynamic behavior of the MIAD algorithm using the diagrams on the next two pages, where X indicates the initial values for x_1 and x_2.


x 2
C
X
C
x 1


x_2
C
X
C
x_1

Question 5

Consider three node network given in the figure below. There are 6 sessions currently using the network: one using the route (1,2,3), one using the route (2,3,1), two sessions using the link (2,3), one using the route (3,1,2), and one using the link (3,1). What is the max-min fair rate allocation in this case? Is the whole network capacity used?



The max-min fair rate allocation is given as follows

$$\begin{array}{lll} x_1 = \frac{1}{4} & x_2 = \frac{1}{4} & x_3 = \frac{1}{4} \\ x_4 = \frac{1}{4} & x_5 = \frac{3}{8} & x_6 = \frac{3}{8} \end{array}$$

Only $\frac{5}{8}$ of the capacity of link (1,2) is being used.