Homework #2

Due Date: 02/21/2017, Tuesday

For both Problems 1 and 2, consider the following algorithm for computing virtual clock values by a VC server for packets in a flow:

$$V(i) = \max\{V(i-1), A(i)\} + s(i)/r \quad \text{for } i = 1, 2, \ldots \tag{1}$$

where $V(0) = 0$, $s(i)$ is size of packet i (in bits), and r is the flow's reserved rate in bits/second.

Problem 1 *(15 points total)*

Prove that for $i = 1, 2, \ldots$

If $A(j+1) - A(j) \leq s(j)/r$, for $j = 1, \ldots, i$ then $V(i) = A(1) + (s(1) + s(2) + \cdots + s(i))/r$.

Problem 2 *(35 points total)*

Suppose the source of the flow is (σ, r) leaky bucket controlled where σ is bucket depth, and r is the flow's reserved rate. Prove the following for $j \geq 1$:

$$V(j) \leq A(j) + \sigma/r. \tag{2}$$

You are required to write an explanation for each step of your proof.

Hints: Prove two cases: (1) $V(j-1) < A(j)$ (this is the easy case) and (2) $V(j-1) \geq A(j)$. The basic idea of a proof can be found in the paper by Goyal, Lam, and Vin (1997) in the second column of page 161. The definitions of GRC values in the paper and VC values herein are the same. Follow the simple notation herein rather than the paper's notation. The key is to understand the meaning of packet k, where k is the largest integer ($\leq j$) in the set defined at the top of the second column of page 161 such that equation (38) holds. Note that equation (38) in the paper is the same as the equation you are asked to prove in Problem 1. What you are asked to prove in Problem 2 is Equation (44) in the paper; the proof in the paper has big gaps.

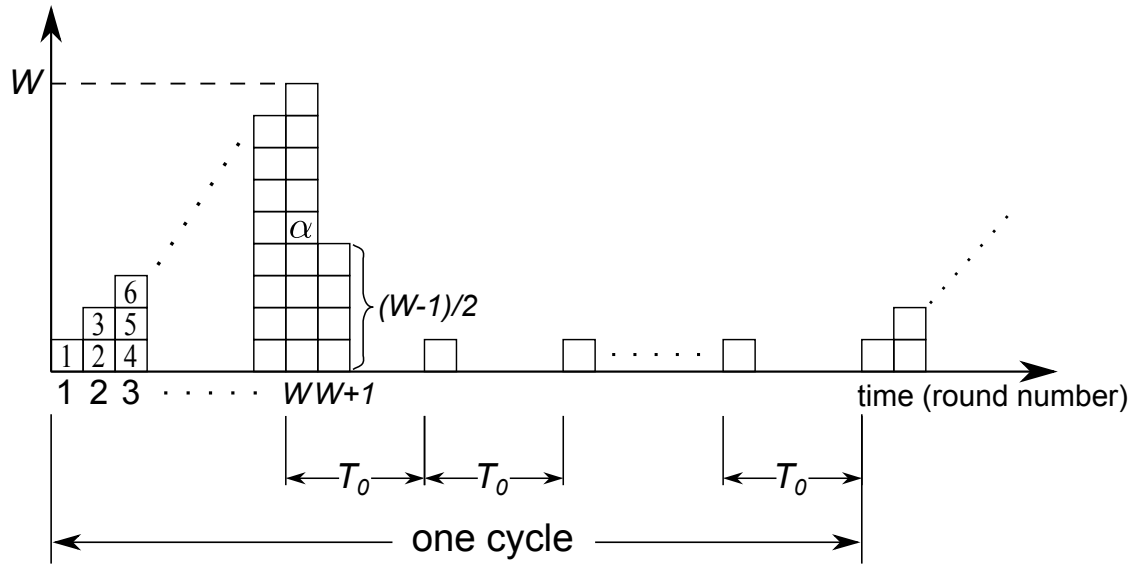

Figure 1: An example of window evolution.

Problem 3 *(40 points total)*

Consider a simplified version of the TCP congestion control protocol as shown in Figure 1 where each square represents one packet sent. The sender does not perform slow start and does not detect triple dupACKs. Each TCP session begins with a window size of 1. Upon receiving an ack, *cwnd* is increased by $\frac{1}{cwnd}$ (as in congestion avoidance) so that *cwnd* increases by 1 per RTT. (There is no need for ssthresh because there is no slow start.) A transmitted packet is timed out with probability p (packet losses are independent). Every lost packet is detected by timeout after T_0 ($>$ 2RTTs) and immediately retransmitted. The window size stays at one while waiting for the ack of a retransmitted packet.

Let W denote the average value of *cwnd* when the first lost packet, packet α (on the average), is sent in the Wth round (on the average) as shown in Figure 1. By the sliding window protocol, $W-1$ more packets are sent following packet α. Since the lost packet can be any packet in the Wth round, the average number of packets sent in the $W+1$st round is $\frac{(W-1)}{2}$.

Unlike TCP congestion control, the timeout duration is fixed at T_0. The receiver provides an ack for every packet received (no delayed ack). The receive window is very large and its impact can be neglected.

(i) Consider an average cycle consisting of one congestion avoidance period and the subsequent period of loss(es) and retransmission(s) as shown in Figure 1. Derive W as a function of p.

(ii) Derive the send rate of the TCP sender as a function of p, T_0 and RTT.