Universitat de Lleida

Exercise 1. Communication Services and Security
Escola Politècnica Superior. Departament d'Informàtica

Problem 1 Let's assume a RED congestion control, having the following parameters:

- `MaxP`=0.4
- `MinTh` = 4
- `MaxTh` = 10

Calculate the following probabilities:

1. Probability of a segment being dropped with a `AvgLen`=8
2. Probability that 3 consecutive segments enter into the queue, assuming that all of them find the same average queue length (`AvgLen`) of 8
3. Same probability as previous point (2) assuming a modified RED congestion control where the probability of a segment being dropped is $\frac{P}{1-\texttt{compt}\cdot P}$, being P the same probability computed at point (1). `compt` is the number of segments that entered into the queue from the last dropped segment. Assume `compt` = 0 for the first segment entering into the queue.

(0.5 points)

Problem 2 Table below shows a *Fair Queuing* (FQ) policy sequence of events, being:

- A, the arrival time of a given segment,
- P, size of the segment.

A	P	F	T_i	T_f
0	4			
0	1			
0.9	1			
1.5	1			
5	2			
8.9	2			
10	4			
10.9	1			
11.9	2			

Fulfill the table, knowing that:

- F is the estimated finishing time of FQ policy,
- T_i is the start transmission time for segment,
- and T_f is its finish transmission time.

(0.5 points)

Problem 3 Write two Python scripts (transmitter and receiver) that using sockets emulates a TCP data interchange (half duplex) according the following rules:

- **General**
  - Use a Karn/Partridge RTT estimator, without Exponential BackOff and $\alpha = 0.8$
  - A Slow Start congestion control (`CWMAX=4`)
  - All transmitted segments have MSS bytes
  - Segment transmission time is 1 s.
  - ACK segment have size 0
  - No headers
  - Propagation delay is 0 s.
- **Transmitter**
  - Segments are numbered starting at 0. Acknowledgment based on segment number (no bytes)
  - Segments with a prime sequence number are considered lost (when transmitted for the first time)
  - Consider that transmitter has always enough information to send
  - Take an initial TimeOut value of 10s.
- **Receiver**
  - Send ACK after 1 s. without a correct reception
  - Send ACK immediately when 3 or more segments in sequence
  - Receiver buffer size: 10 MSS
  - Consider always an infinite `AdvertisedWindow`
- **To deliver**
  - A short report including:
    - Comments on the main aspects of your encoding
    - Description of the segment format for data and ACK segments
    - The following table from your output code:

      Time (s) | Event | Eff.Win. (MSS) | cwnd (MSS) | RTT (s) | sRTT (s) | TOut (s)

      during the first 200 seconds.
    - Plots of `cwnd` and sRTT as a function of time
  - The Python code for transmitter and receiver

Some **hints**:

- Use `socket` lib. Datagram may be a good option
- Use `threading` and/or `multiprocessing` and `time` libs to implement TimeOut timers, scheduled acks, ...
- Use `multiprocessing.Manager()` to deal with global structures
- Use `sympy` for primality test

(4 points)