Part B Congestion control

For each TCP flow:
(1) Print the first ten congestion window sizes (or till the end of the flow, if there are less than five congestion windows). You need to decide whether the congestion window should be estimated at the sender or the receiver and explain your choice. Mention the size of the initial congestion window. You need to estimate the congestion window size empirically since the information is not available in the packet. Comment on how the congestion window size grows. Remember that your estimation may not be perfect, but that is ok. Congestion window sizes are typically estimated per RTT.

The congestion window should be estimated at the sender. Because TCP will take the arrival of acknowledgments as an indication that all is well, and will use acknowledgments to increase its congestion window size (and hence its transmission rate), and TCP uses acknowledgments to trigger (or clock) its increase in congestion window size.

Here the congestion window is calculated by counting the packets sent during each RTT(from avg_rtt calculated in PartA, we take 0.08s here), multiplied by MSS(here are 1460 bytes according to the TCP header). The results are given as follows:

```
======== Part B (1) ========
src port: 43498, MSS: 1460 bytes
packet count of each RTT: [10, 20, 33, 45, 66, 101, 135, 203, 270, 405]
congestion window size in each RTT: [14600, 29200, 48180, 65700, 96360, 147460, 197100, 296380, 394200, 591300]
increase rate: [1, 2.0, 1.65, 1.36, 1.47, 1.53, 1.34, 1.5, 1.33, 1.5]

src port: 43500, MSS: 1460 bytes
packet count of each RTT: [10, 20, 33, 45, 66, 101, 135, 203, 271, 404]
congestion window size in each RTT: [14600, 29200, 48180, 65700, 96360, 147460, 197100, 296380, 395660, 589840]
increase rate: [1, 2.0, 1.65, 1.36, 1.47, 1.53, 1.34, 1.5, 1.33, 1.49]

src port: 43502, MSS: 1460 bytes
packet count of each RTT: [10, 20, 33, 44, 66, 90, 135, 181, 147]
congestion window size in each RTT: [14600, 29200, 48180, 64240, 96360, 131400, 197100, 264260, 214620]
increase rate: [1, 2.0, 1.65, 1.33, 1.5, 1.36, 1.5, 1.34, 0.81]
```

1) For flow1 and flow2, the increase rate is always > 1 for the first 10 RTT, though it is not like the Slow Start we have seen that grows exponentially, we still can say it is at Slow Start.
2) for flow3, the increase rate > 1 for the first 8 RTT, however, the cwnd is decreased for the 9th RTT, then we can assume dup ACKs event occurs at the 8th RTT, thus it is at the fast Recovery mode.

(2) Compute the number of times retransmission occurred due to triple duplicate ack and the number of times retransmission occurred due to timeout (as before, determine if you need to do it at the sender or the receiver).

The duplicate acks are determined at the receiver side, and retransmission is determined at the sender. First, we calculate the number of times of ack x on the receiver side, and seq x in the sender is sent separately. Then, if total(seq of x) > 1, then increase total retransmission times by 1, otherwise, if total(ack of x) > 1, then increase the triple duplicate ack by 1.

```
======== Part B (2) ========
src port: 43498
retransmitted by dupACKs: 0
retransmitted by timeout: 3

src port: 43500
retransmitted by dupACKs: 2
retransmitted by timeout: 94

src port: 43502
retransmitted by dupACKs: 0
retransmitted by timeout: 0
```