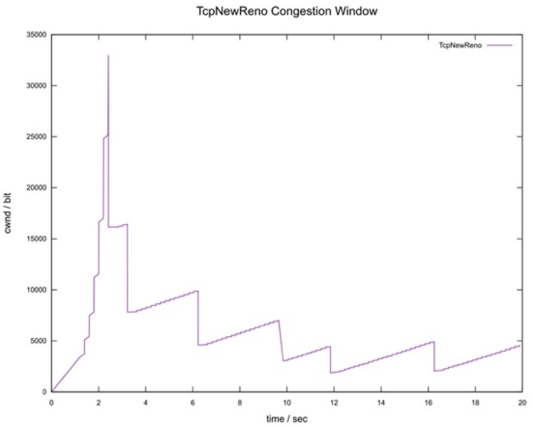
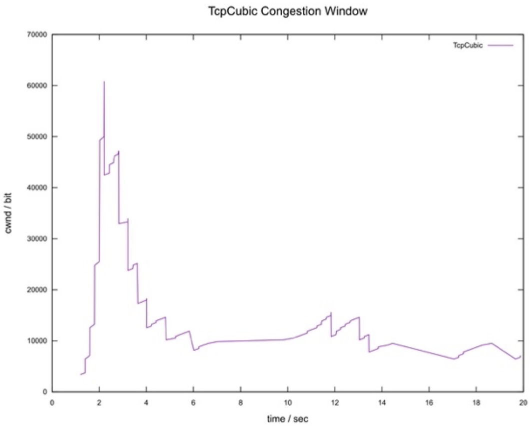
Part1a





When a packet is lost in TcpNewReno， TcpNewReno uses a fast recovery algorithm. It will first set half of the current CWND as ssthresh and as new CWND, thus, it skips a slow start and goes directly to the congestion avoidance algorithm.

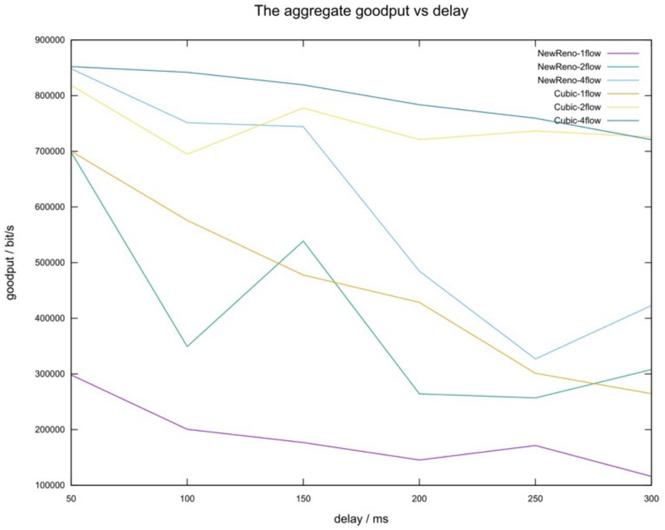
The idea of the congestion avoidance algorithm is to slowly increase the congestion window cwnd, that is, increase the sender's congestion window by 1 cwnd in 1 RTT, instead of doubling it. In this way, the congestion window grows slowly and linearly, which is much slower than the slow start algorithm.

When a packet is lost in TcpCubic, it also enters congestion avoidance from fast recovery. TcpCubic needs to get Wmax, which is the current congestion window by a factor of β where β is a window decrease constant. It starts to increase the window using the concave profile of the cubic function until the window size becomes Wmax. After that, the cubic function becomes a convex profile, and the window starts to grow. Thus, the window size remains almost constant around Wmax, forming a plateau where network utilization is the highest, and in a steady state.

The goodput of TcpNewReno is 221322 bits/s and TcpCubic is 497330 bits/s.

To sum up, the TcpCubic congestion window remains almost constant around Wmax for a long period, which causes higher network utilization. Consistent with the Lab2 Part1a results.

Part1b

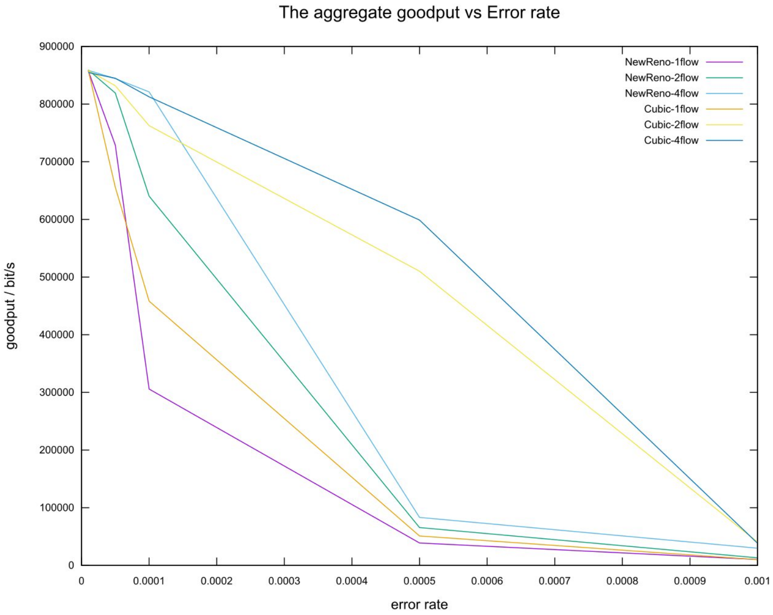


For TcpNewReno, the longer delay will increase the RTT and more packets loss which will slow down the growth of the congestion window and lead to lower goodput.

For TcpCubic, Cubic's congestion window is not affected by RTT. Thus, TcpCubic is less affected by transmission delay and maintains high goodput.

For both TcpNewReno and TcpCubic, increasing the number of flows can increase the aggregate goodput.

Part1c



The results of Lab2 part1c show that the goodput will drop rapidly after the error rate exceeds 0.0001 for both TcpNewReno and TcpCubic. On average, TcpCubic performs better and has a higher goodput than TcpNewReno at the same error rate.

Increasing the number of flows also can increase the aggregate goodput for both TcpCubic and TcpNewReno.