Lab 3: Results

Door

Filip Skalka en Mark Jansen
Networks and Security
26 September 2022

3.1 Answer:

Answer to question of task 3.1:

the plot indicates that throughout the majority of the time (during the congestion control phase of the algorithm) the ratio of CWND and RTT is roughly constant.

3.2.1 Answer:

Because the Congestion Window (CWND) floats between 50% of the maximum CWND size and 100%. This means that the mean of the congestion window size is 75% and that's why it was incorporated in the formula.

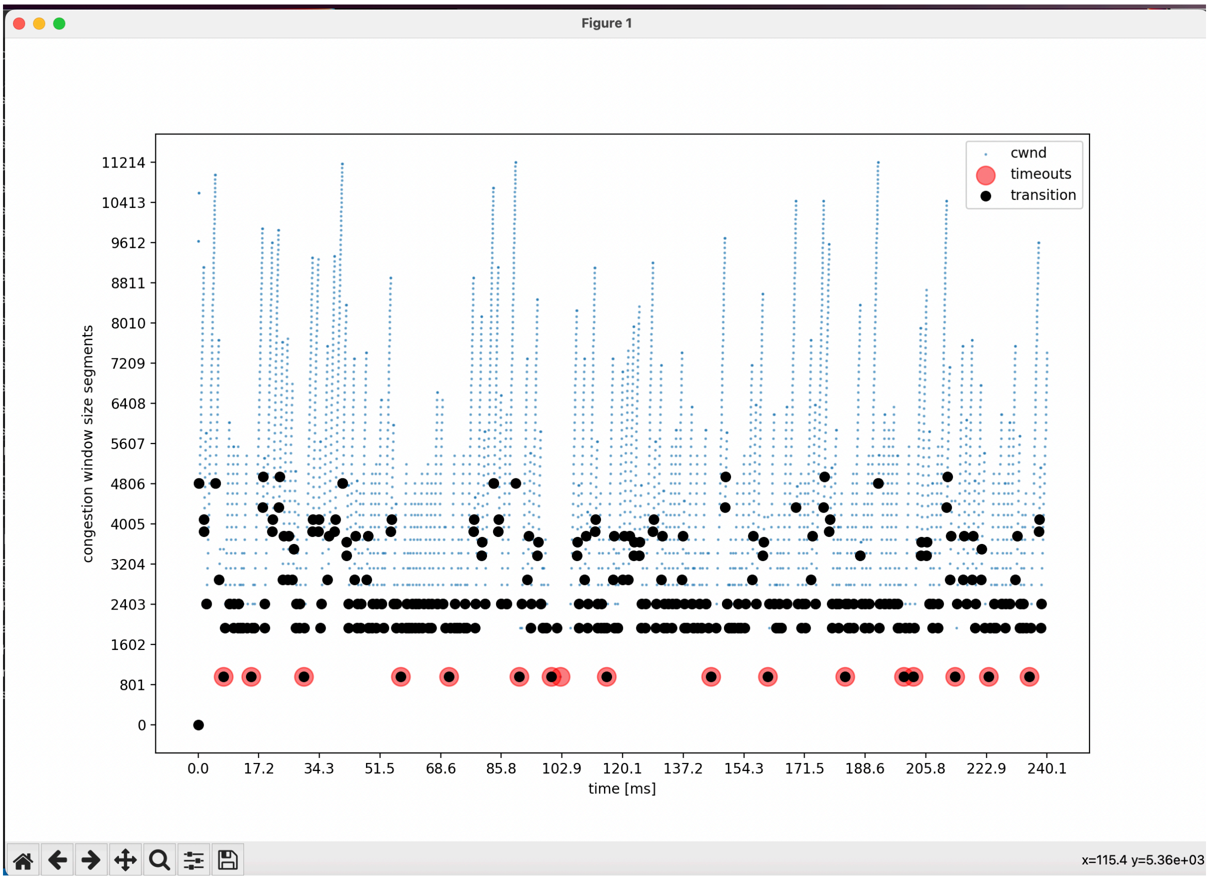
3.2.2 Answer:

The approximation might be off because in computing the actual average we are considering only the values of cwnd and rtt where the timestamps match in both trace files. In the approximation there is no such requirements which means we can take an arbitrary cwnd before the loss event value which can make the difference and we are also considering all of the RTT values which we are not doing in the actual average computation.

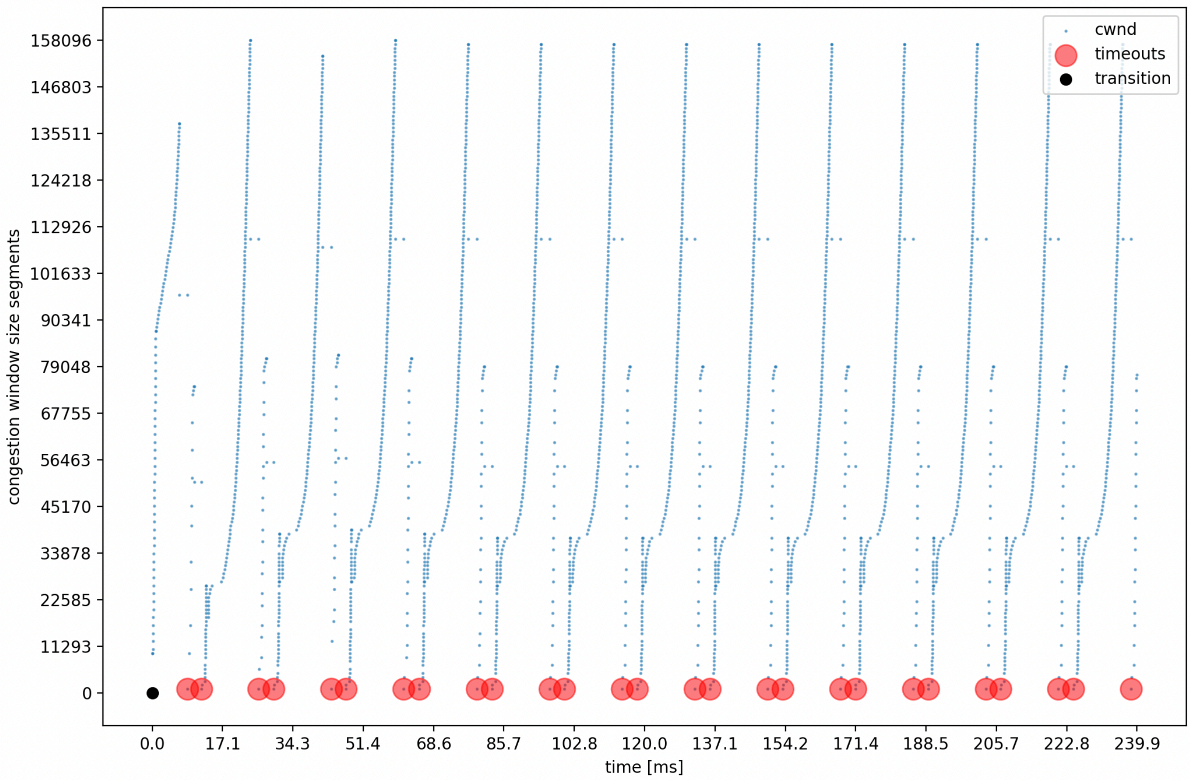
3.3 Answer:

The reason for the dramatic effect, which is also greatly visible in the chart, are the timeouts that occur when the acknowledgement has not been received within the timeout interval due to packet loss.

4.1 Answer:



TCP New Reno



TCP Cubic

4.1.1 Answer:

The difference between TCP Cubic and TCP New Reno is the way they behave when they hit their threshold. Cubic stays near the threshold at first and ramps up exponentially and Reno's congestion avoidance phase increases linearly.

4.1.2 Answer:

TCP Cubic is preferred when the network is more congested and you wanna take advantage of the new and available bandwidth.

4.2.1 Answer:

Two delay signals are commonly used for congestion control, round-trip time TRTT and one-way delay TOWD. The ultimate goal is to obtain a measure of the queueing delay Tq between the sender and receiver, but both of these measures include delays within the sender, within the receiver, and the propagation delays along the path. Most delay-based TCP variants use TRTT, as then only the TCP sender needs to be modified.

4.2.2 Answer:

FAIRNESS TCP Vegas does not co-exist well with other congestion control algorithms because it is the fastest to detect congestion and throttle the connection, before other algorithms that use packet loss as the main indication

FAIRNESS TCP Vegas does not co-exist well with other congestion control algorithms because it is the fastest to detect congestion and throttle the connection, before other algorithms that use packet loss as the main indication

Bijlage voor commands:

Task 4.1: ./ns3 run tcp-variants-comparison --command-template="%s --transport_prot=TcpCubic --bandwidth=1Mbps --tracing=true --prefix_name=Task-4.1 --mtu=1024 --duration=240 —sack=false”

Task 1.2: ./ns3 run tcp-variants-comparison --command-template="%s --transport_prot=TcpNewReno --bandwidth=1Mbps --tracing=true --prefix_name=Task-1.2 --mtu=1024 --duration=240 --sack=false"

Task 3.3: ./ns3 run tcp-variants-comparison --command-template="%s --transport_prot=TcpCubic --error_p=0.03 --bandwidth=1Mbps --tracing=true --prefix_name=Task-4.1 --mtu=1024 --duration=240 --sack=false"

Task 3.1: ./ns3 run tcp-variants-comparison --command-template="%s --transport_prot=TcpNewReno --tracing=true --prefix_name=Task-3.1 --duration=480"

Task 1.1: ./ns3 run tcp-variants-comparison --command-template="%s --transport_prot=TcpNewReno --tracing=true --prefix_name=Task-1.1 --duration=240"