1 Basic Tests

Each of these tests on the Bene simulator was run with a link throughput of 10 Mbps, and a propagation delay of 10 ms.

1.1 Small File

The smaller file, `test.txt`, was transferred with a window size of 3000 bytes. It's clear, from the data below, that the loss rate has an extremely high effect on the total time needed to transmit the data.

0% loss rate:

```
0.064 n1 (2) received ACK from 1 for 9000
0.064 n1 cannot send more data
0.0732 n2 (1) received TCP segment 9000-9999 from 2
0.0732 n2 receive buffer now has 9000-9999
0.0732 Received 1000 bytes
0.0732 n2 (1) sending TCP ACK to 2 for 10000
0.0832 n1 (2) received ACK from 1 for 10000
0.0832 n1 cancel timer
0.0832 n1 cannot send more data
 === File transferred successfully ===
```

10% loss rate:

```
2.1064 n1 (2) received ACK from 1 for 9000
2.1064 n1 cannot send more data
2.1156 n2 (1) received TCP segment 9000-9999 from 2
2.1156 n2 receive buffer now has 9000-9999
2.1156 Received 1000 bytes
2.1156 n2 (1) sending TCP ACK to 2 for 10000
2.1256 n1 (2) received ACK from 1 for 10000
2.1256 n1 cancel timer
2.1256 n1 cannot send more data
 === File transferred successfully ===
```

20% loss rate:

```
3.1456 n1 (2) retransmission timer fired, sequence 8000
3.1456 n1 (2) sending TCP segment 8000-8999 to 1
3.1564 n2 (1) received TCP segment 8000-8999 from 2
3.1564 n2 receive buffer now has 8000-9999
3.1564 Received 2000 bytes
3.1564 n2 (1) sending TCP ACK to 2 for 10000
3.1664 n1 (2) received ACK from 1 for 10000
3.1664 n1 cancel timer
3.1664 n1 cannot send more data
 === File transferred successfully ===
```

50% loss rate:

```
16.1664 n1 (2) retransmission timer fired, sequence 8000
16.1664 n1 (2) sending TCP segment 8000-8999 to 1
16.1772 n2 (1) received TCP segment 8000-8999 from 2
16.1772 n2 receive buffer now has 8000-9999
16.1772 Received 2000 bytes
16.1772 n2 (1) sending TCP ACK to 2 for 10000
16.1872 n1 (2) received ACK from 1 for 10000
16.1872 n1 cancel timer
16.1872 n1 cannot send more data
 === File transferred successfully ===
```

1.2 Large File

The larger file, `internet-architecture.pdf`, was transferred with a window size of 10,000 bytes. Interestingly enough, when exposed to a 50% loss rate, the effect when compared to the 0% loss rate is not as drastic as it was for the large file. This effect is also generally reproducible. My guess is that with a larger amount of data to work with, and a larger window, an error that occurs in the middle will still likely not largely affect the end.

0% loss rate:

```
1.0824 n1 (2) received ACK from 1 for 512000
1.0824 n1 cannot send more data
1.0832 n1 (2) received ACK from 1 for 513000
1.0832 n1 cannot send more data
1.084 n1 (2) received ACK from 1 for 514000
1.084 n1 cannot send more data
1.08442 n1 (2) received ACK from 1 for 514520
1.08442 n1 cancel timer
1.08442 n1 cannot send more data
 === File transferred successfully ===
```

50% loss rate:

```
361.4928 n1 (2) retransmission timer fired, sequence 511000
361.4928 n1 (2) sending TCP segment 511000-511999 to 1
361.5036 n2 (1) received TCP segment 511000-511999 from 2
361.5036 n2 receive buffer now has 511000-514519
361.5036 Received 3520 bytes
361.5036 n2 (1) sending TCP ACK to 2 for 514520
361.5136 n1 (2) received ACK from 1 for 514520
361.5136 n1 cancel timer
361.5136 n1 cannot send more data
 === File transferred successfully ===
```

2 Fast Retransmit

Both of these tests transferred the large file and a 10,000 byte window. This was easier than expected to implement.

2.1 With Fast Retransmit

With fast retransmit, I noticed quite a bit of time was shaved off. After analyzing individual logs, I noticed that fast retransmit was much more likely to trigger in the beginning or middle of the transmission rather than at the end. This seemed to mostly be due to the fact that fewer ACK packets will be received when the rest of the file has been transferred, greatly decreasing the chance of many duplicate ACK packets.

0% loss rate:

```
1.0824 n1 (2) received ACK from 1 for 512000
1.0824 n1 cannot send more data
1.0832 n1 (2) received ACK from 1 for 513000
1.0832 n1 cannot send more data
1.084 n1 (2) received ACK from 1 for 514000
1.084 n1 cannot send more data
1.08442 n1 (2) received ACK from 1 for 514520
1.08442 n1 cancel timer
1.08442 n1 cannot send more data
 === File transferred successfully ===
```

20% loss rate:

```
47.80562 n2 (1) sending TCP ACK to 2 for 514520
47.80562 2 dropped packet due to random loss
48.79482 n1 (2) retransmission timer fired, sequence 513000
48.79482 n1 (2) sending TCP segment 513000-513999 to 1
48.80562 n2 (1) received TCP segment 513000-513999 from 2
48.80562 n2 (1) sending TCP ACK to 2 for 514520
48.81562 n1 (2) received ACK from 1 for 514520
48.81562 n1 cancel timer
48.81562 n1 cannot send more data
 === File transferred successfully ===
```

2.2 Without Fast Retransmit

Obviously, no difference was noticed in the 0% loss rate scenario.

0% loss rate:

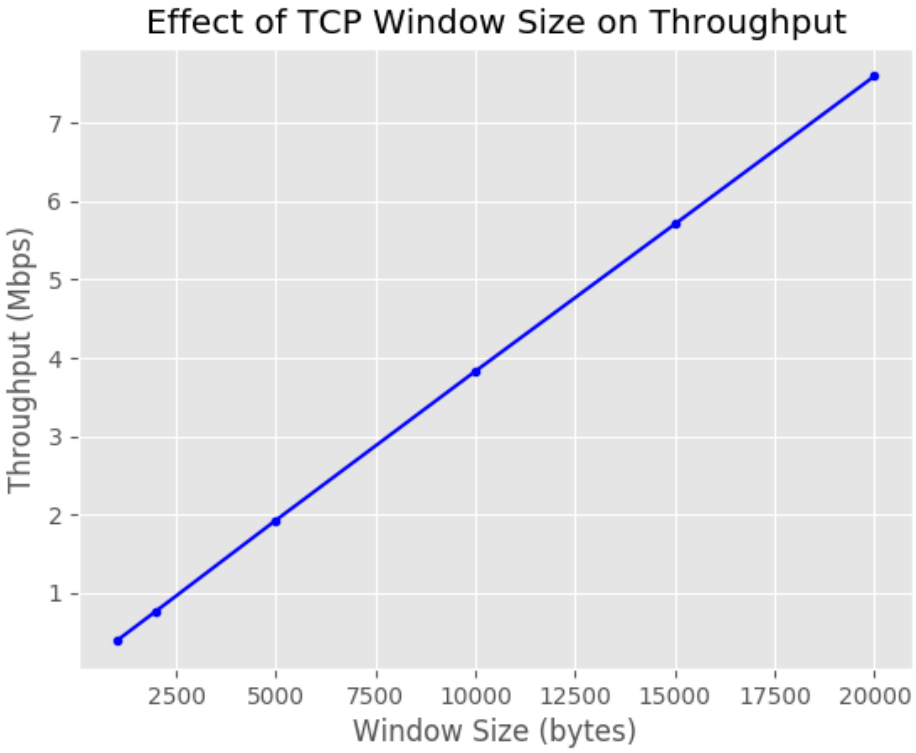
```
1.0824 n1 (2) received ACK from 1 for 512000
1.0824 n1 cannot send more data
1.0832 n1 (2) received ACK from 1 for 513000
1.0832 n1 cannot send more data
1.084 n1 (2) received ACK from 1 for 514000
1.084 n1 cannot send more data
1.08442 n1 (2) received ACK from 1 for 514520
1.08442 n1 cancel timer
1.08442 n1 cannot send more data
 === File transferred successfully ===
```

20% loss rate:

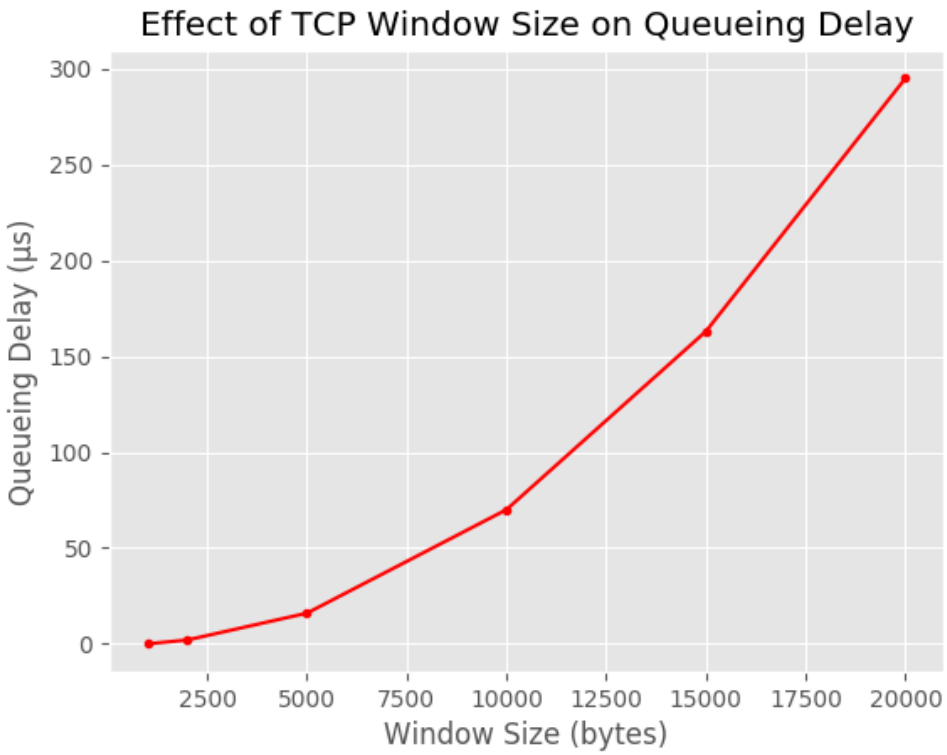
```
63.8052 n2 (1) sending TCP ACK to 2 for 514520
63.8052 2 dropped packet due to random loss
64.7944 n1 (2) retransmission timer fired, sequence 510000
64.7944 n1 (2) sending TCP segment 510000-510999 to 1
64.8052 n2 (1) received TCP segment 510000-510999 from 2
64.8052 n2 (1) sending TCP ACK to 2 for 514520
64.8152 n1 (2) received ACK from 1 for 514520
64.8152 n1 cancel timer
64.8152 n1 cannot send more data
 === File transferred successfully ===
```

3 Experiments

Using a bandwidth of 10 Mbps, a propagation delay of 10 ms, a queue size of 100, and a loss rate of 0%, a transfer of the large file was performed with varying window sizes. The goal was to discover the effect of the TCP window size on throughput and average queuing delay.
As seen below, the window size appears to have a linear effect on the throughput of the connection. The maximum throughput of the link was 10 Mbps, and as the window size grew closer to that, so the TCP throughput was brought closer to the maximum.



However, the average queuing of packets did not have the same behavior. As the window size (and throughput) went up, the queuing delay also went up–but more in a quadratic fashion. In the first window sizes, the queuing delay was next to negligible. At first, I thought my method of calculating queuing delay was flawed, but then realized it was simply nothing for windows this small. Even still, it is nearly negligible. In the end, the highest average queuing delay, for a 20,000-byte window size, was 0.3 milliseconds.
I also ran into another situation that was rather baffling. It seems that, although the queue size was explicitly defined for this scenario and nothing else, it had no effect on my outcome. Not once was a link's queue full enough to cause packets to be dropped.


Effect of TCP Window Size on Queueing Delay
Queueing Delay (µs)
300
250
200
150
100
50
0
2500
5000
7500
10000
12500
15000
17500
20000
Window Size (bytes)