Assignment 3 · CS32305

Introduction to Computer Networks

# Simulation of the TCP Congestion Control Algorithm

Abhijit Hota · CE19B033

April 25, 2022

# Introduction

The report should explain how different factors influence the $CW$ change over the duration of the session.

## Notations & Assumptions

We repeat the notation and assumptions in the problem statement for the sake of completeness. We also add some of our own notations to ease the prose.

| Notation | Description | Range |
|---|---|---|
| $CW$ | Congestion window | |
| $K_i$ | Initial CW | $1.0 \leq K_i \leq 4.0$ |
| $K_m$ | Multiplier of CW during the exponential growth phase | $0.5 \leq K_m \leq 2.0$ |
| $K_n$ | Multiplier of CW during the linear growth phase | $0.5 \leq K_n \leq 2.0$ |
| $K_f$ | Multiplier when a timeout occurs | $0.1 \leq K_f \leq 0.5$ |
| $P_s$ | Probability of receiving the `ACK` packet for a given segment before its timeout occurs | $0.0 \leq P_s \leq 1.0$ |
| $T$ | Number of segments to be sent for the emulation | |

## Methodology

The task was to assess the congestion control algorithm of TCP with various parameters. A total of 6 changeable parameters can be given to the simulation which affect the congestion window size.
The following arguments were given to us.

$K_i \in \{1, 4\}, K_m \in \{1, 1.5\}, K_n \in \{0.5, 1\}, K_f \in \{0.1, 0.3\}, P_s \in \{0.99, 0.9999\}$

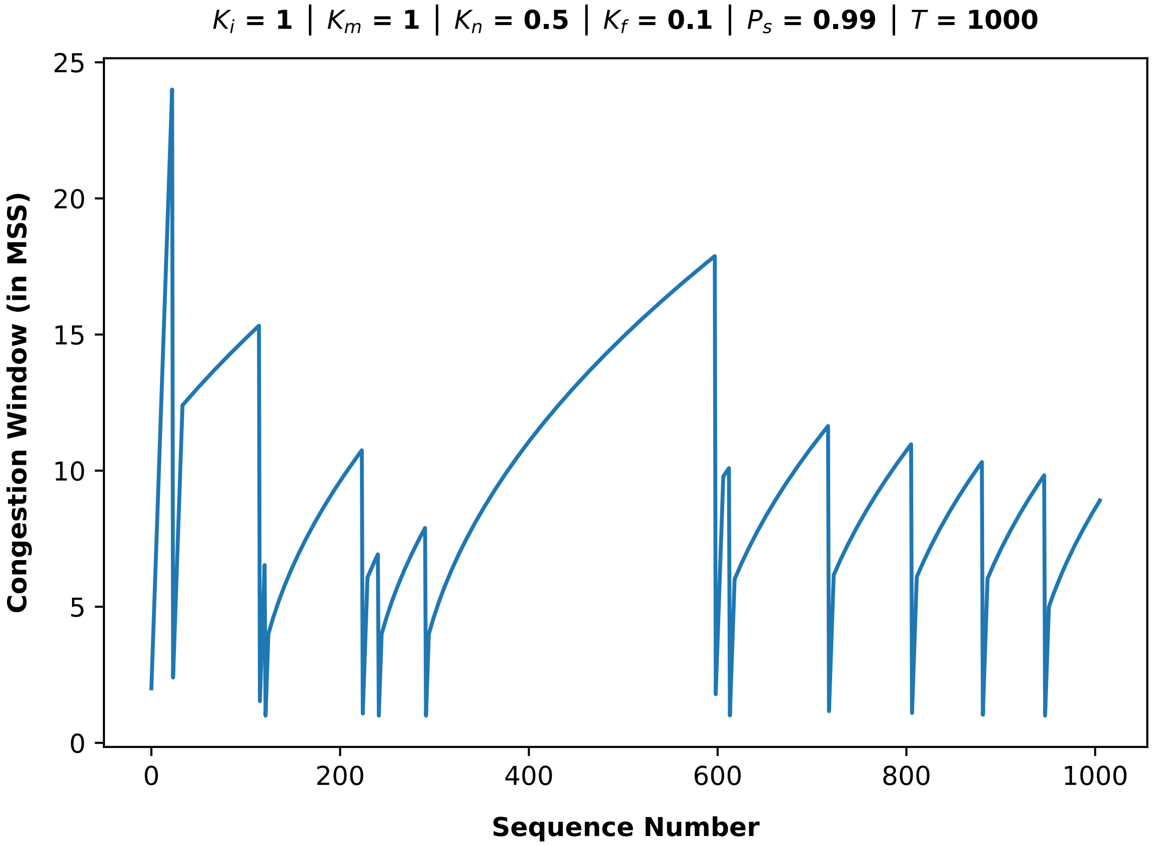
All combination makes a total of 32 possible graphs. A value of 1000 was taken for $T$. A Python script was created to iterate over all possible values and output the files to the path `/figures/*.svg` and `/figures/*.log`.

The Python script can be found in the code repository with the name `simulate_from_parameters.py`

# Results

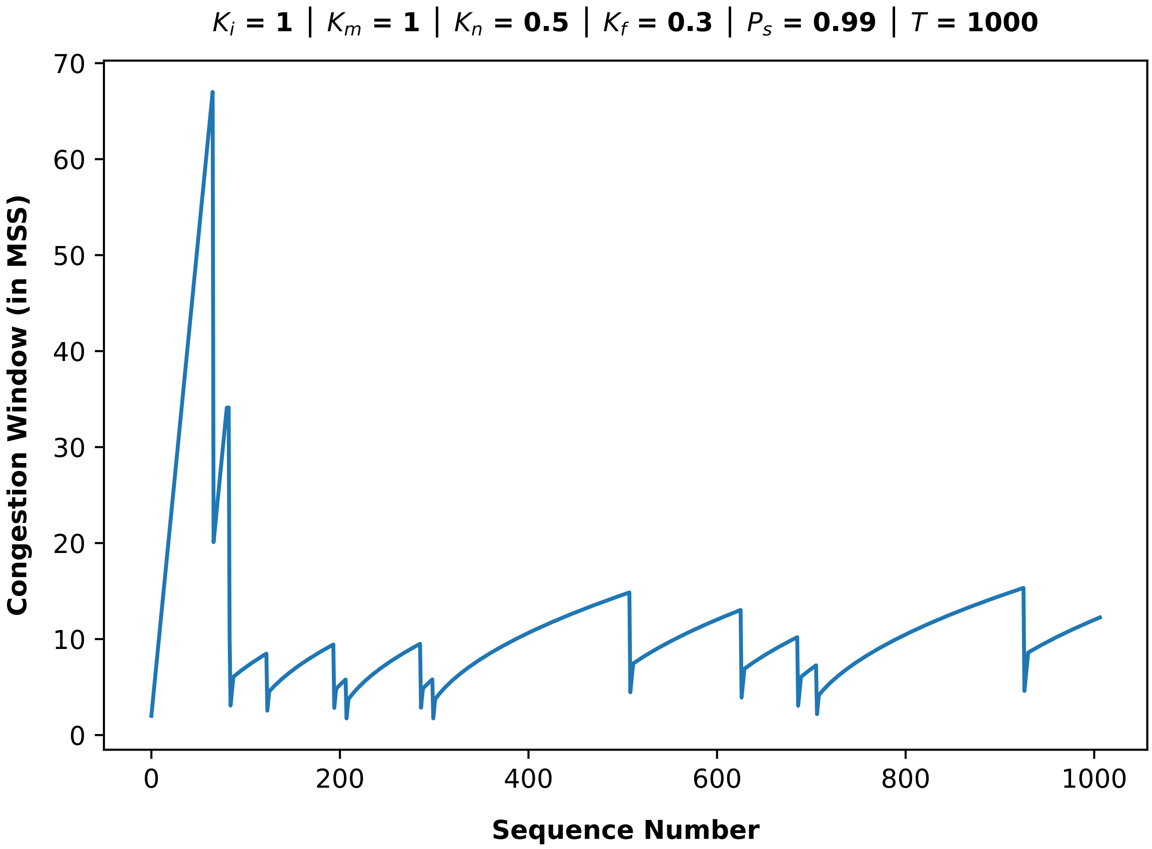
## $P_s = 0.99$

When the probability is 0.99, we see that we get the expected sawtooth behavior from the graphs. We go over the graphs one by one:



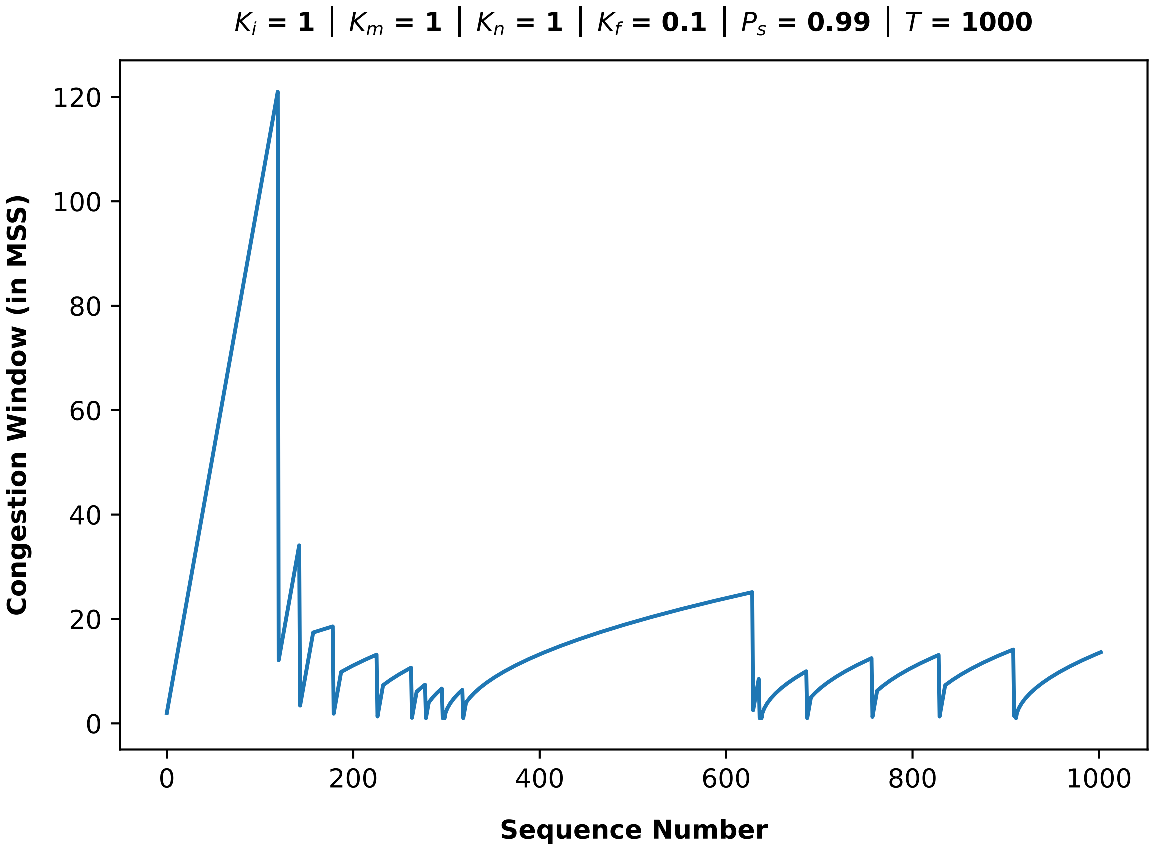

Here, we see that the highest CW value is low $\approx 24$. This can be the result of a lower $K_f$ value. A low $K_f$ value decreases the CW by a 90% and throws it back into slow start, every time a timeout occurs.

Also, notice the near-prefect sawtooth shape from sequence number $\approx 600$ to the end. The CW falls back to the same value every timeout.



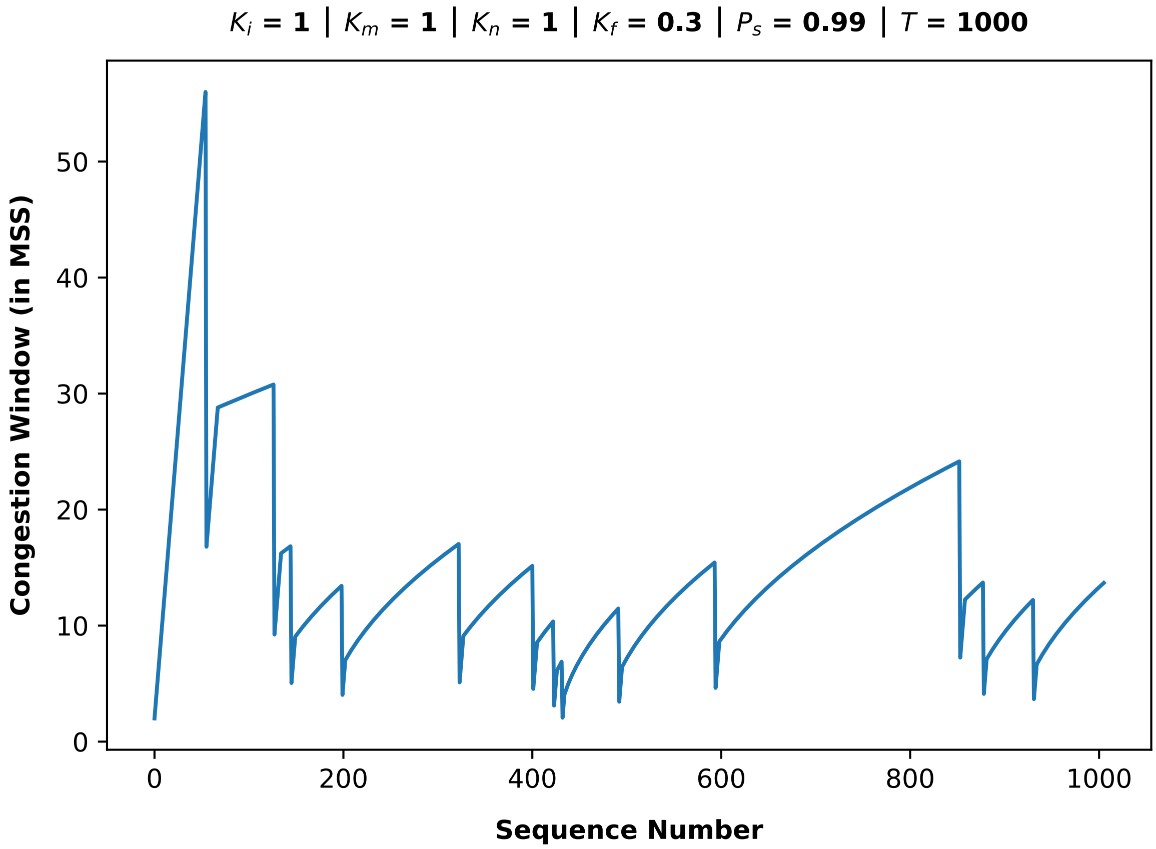

While we saw a few near perfect sawtooth shape in the previous graph, this graph tells a different story. This has bigger sawtooth shapes followed by smaller ones lasting for lesser sequences. This process repeats from sequence number $\approx$ 160 to 300 and starts again from there to 700. Ths can be attributed to a higher $K_f$ value which decreases the CW value by 70%.

We also see a higher maximum CW value of $\approx 68$.

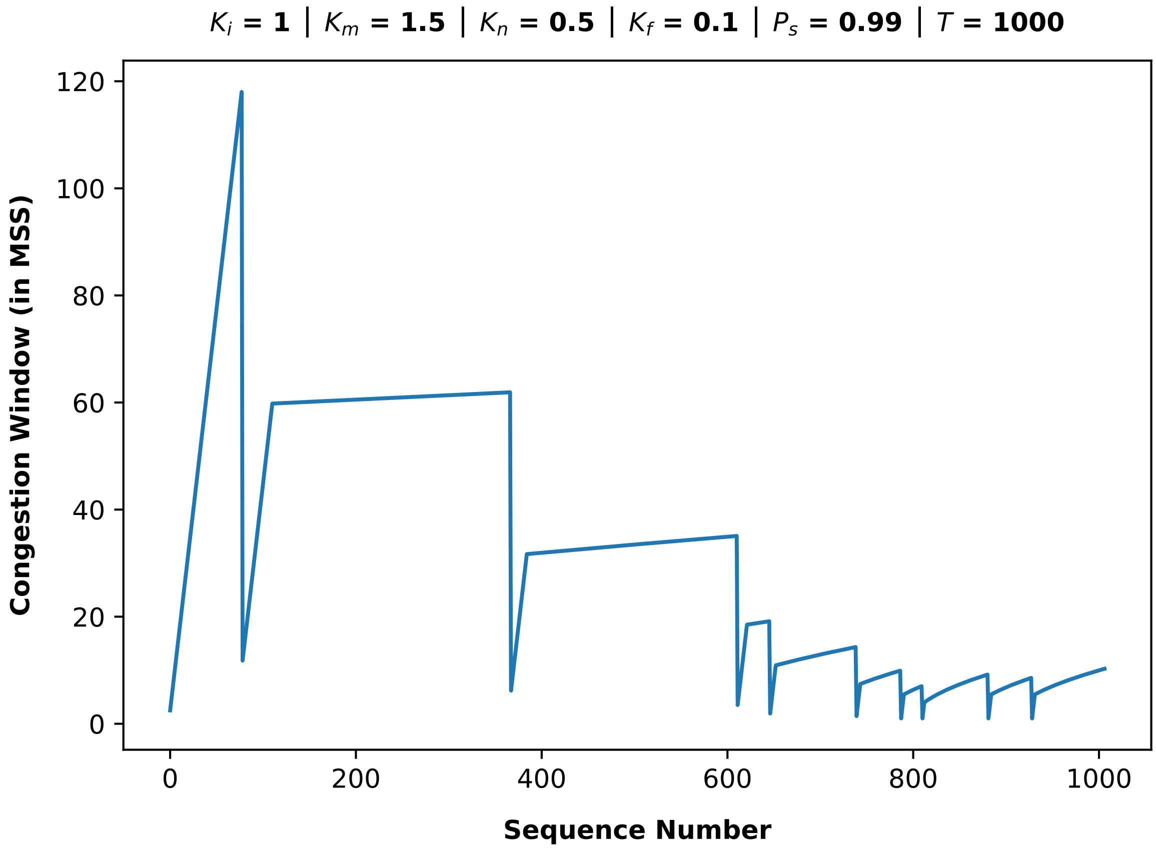



Here, we increase $K_n$ to 1 and observe that the graphs becomes more steeper (higher slope) during the phase when they are more than the congestion threshold. This is inline with the formula given for the linear growth phase.

Also, observe the clean sawtooth shape towards the end.

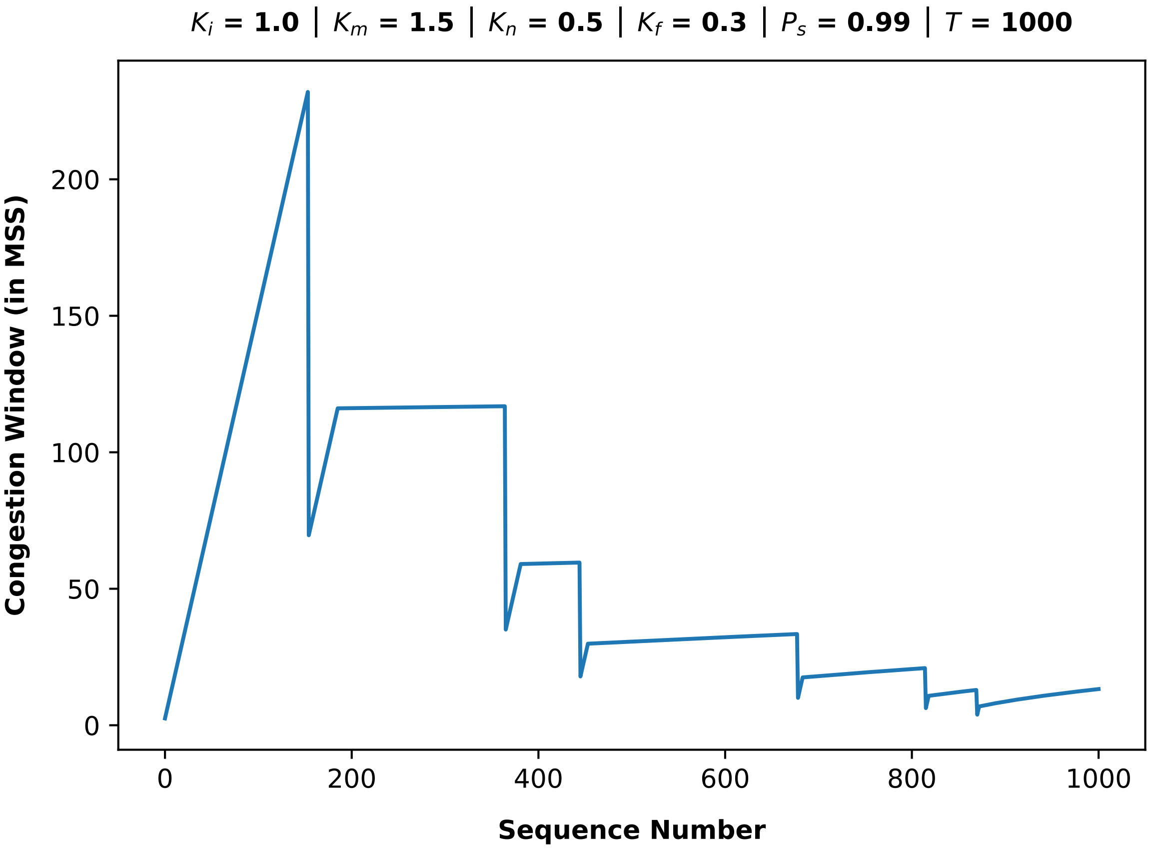



This tells the same story of higher valued slopes owing to a higher $K_n$ value and smaller sawteeth following the larger ones which are a result of a higher $K_f$ value.

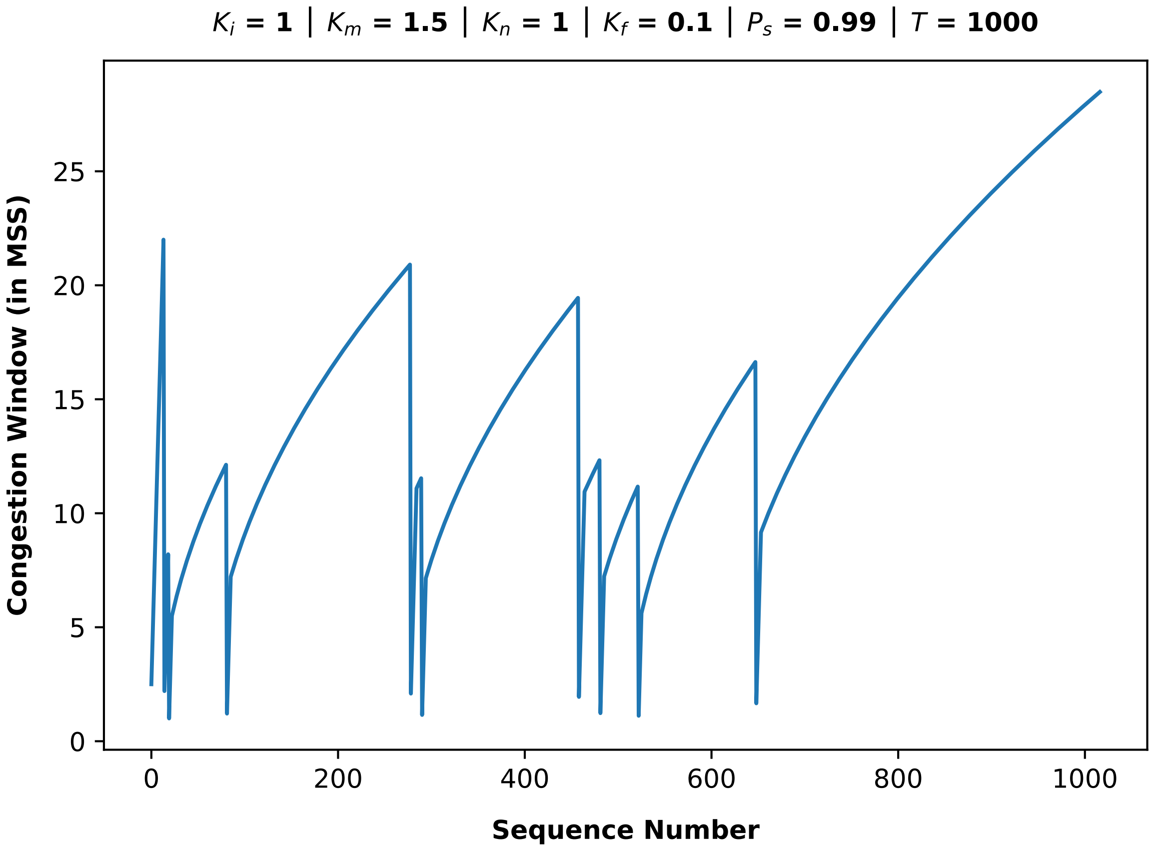



The maximum value of the CW in this graph is rather high. The high $K_m$ value is responsible for this.

This graph is observed to have a very flat slope in the linear phase. It owes it to the low $K_n$ value and the initial maxima of CW.

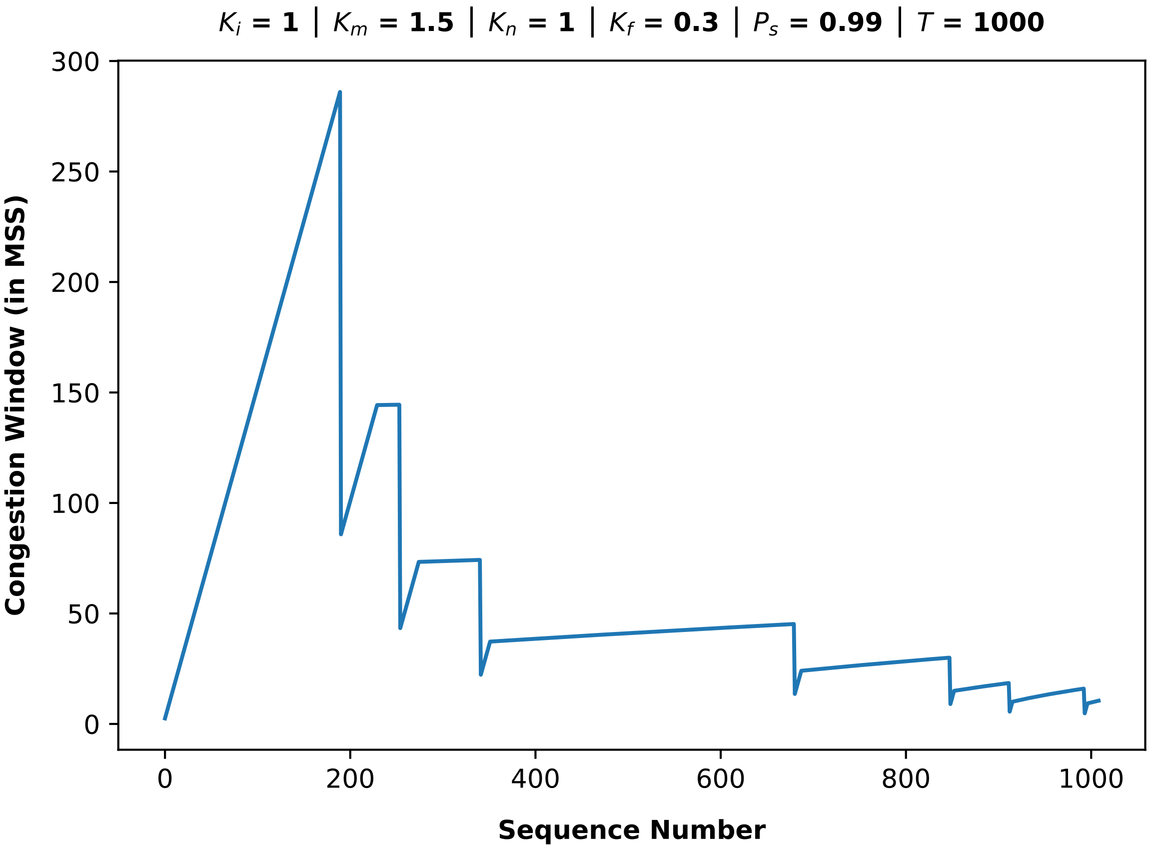



Similar observation as above can be made to this graph.



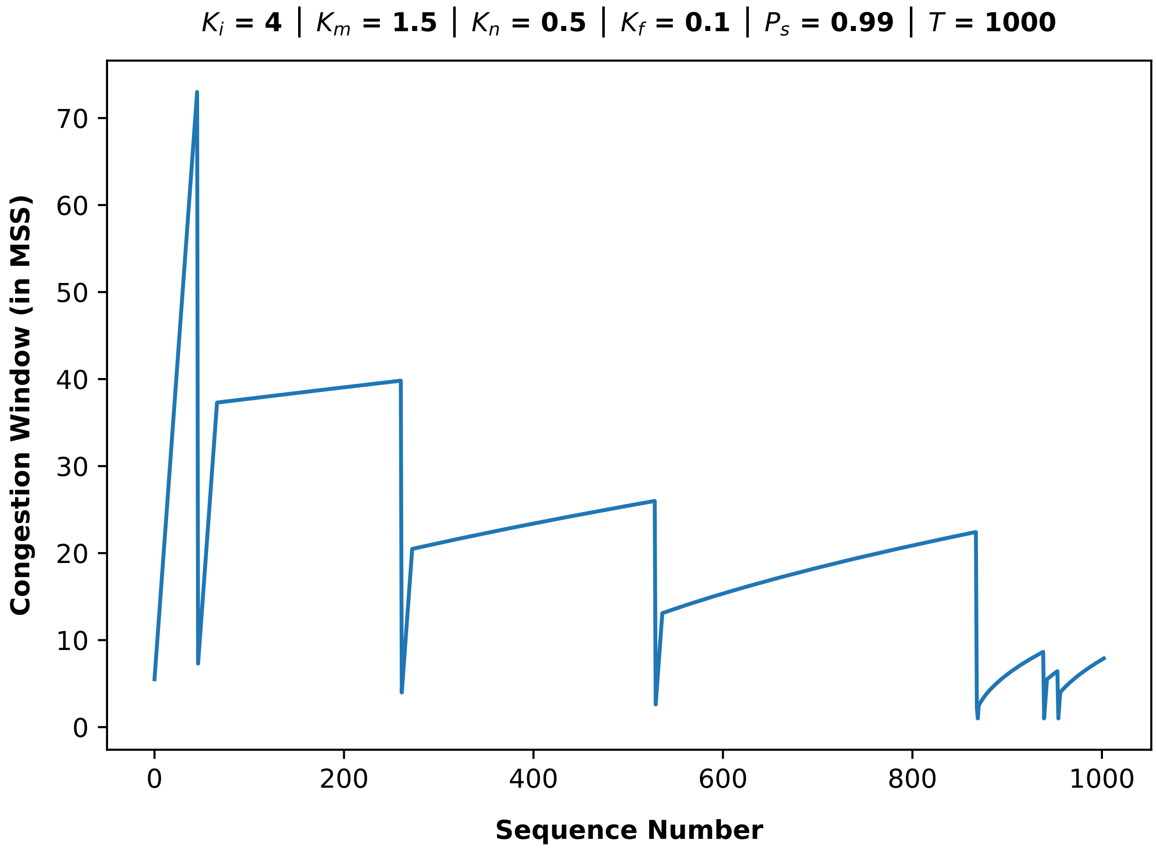

This graph is an interesting case of chance. Notice how the slopes looks steep but they are actually not if you calculate the actual slope value. The reason being the highest CW values is very low.

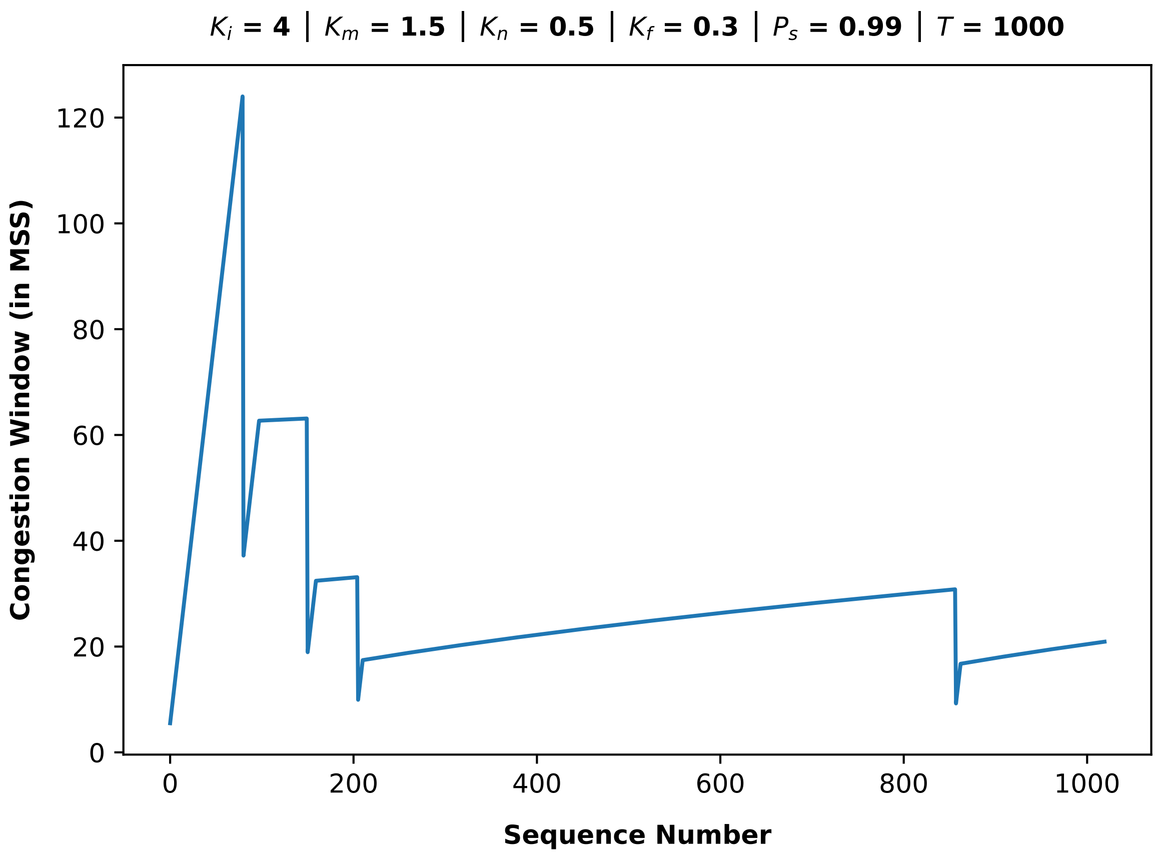
This graph (and a couple of other graphs below) tell us that the shape is dependent on the initial bump of the CW value. Here, the timeout occurs at an early stage which hinders the growth of CW and gives this shape to the graph.



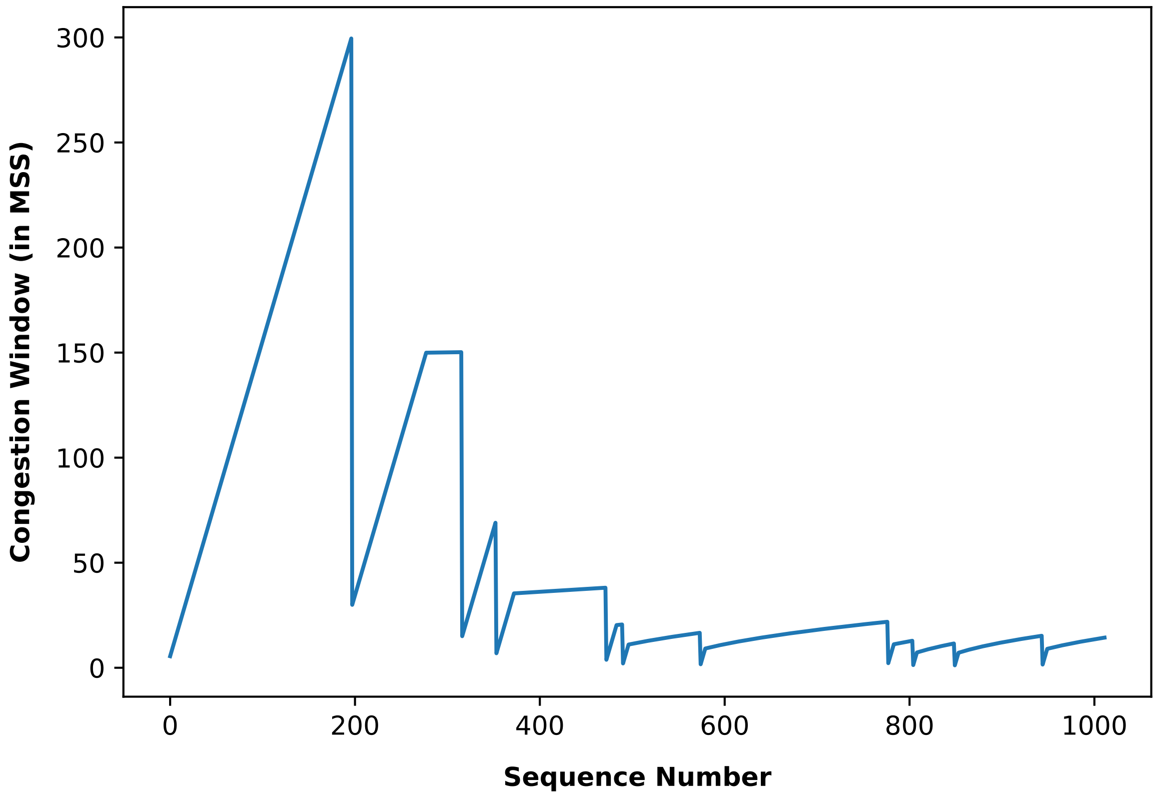

This graph has a high initial CW bump and capped peaks, owing to the high $K_f$ value. The CW doesn't go all the way down and hence the slow start can go higher up to the threshold without getting timed out quickly.

For the following graphs, similar qualitative analytics can be made. The $K_i$ is made higher in the following graphs which make it more probable to have higher CW values.

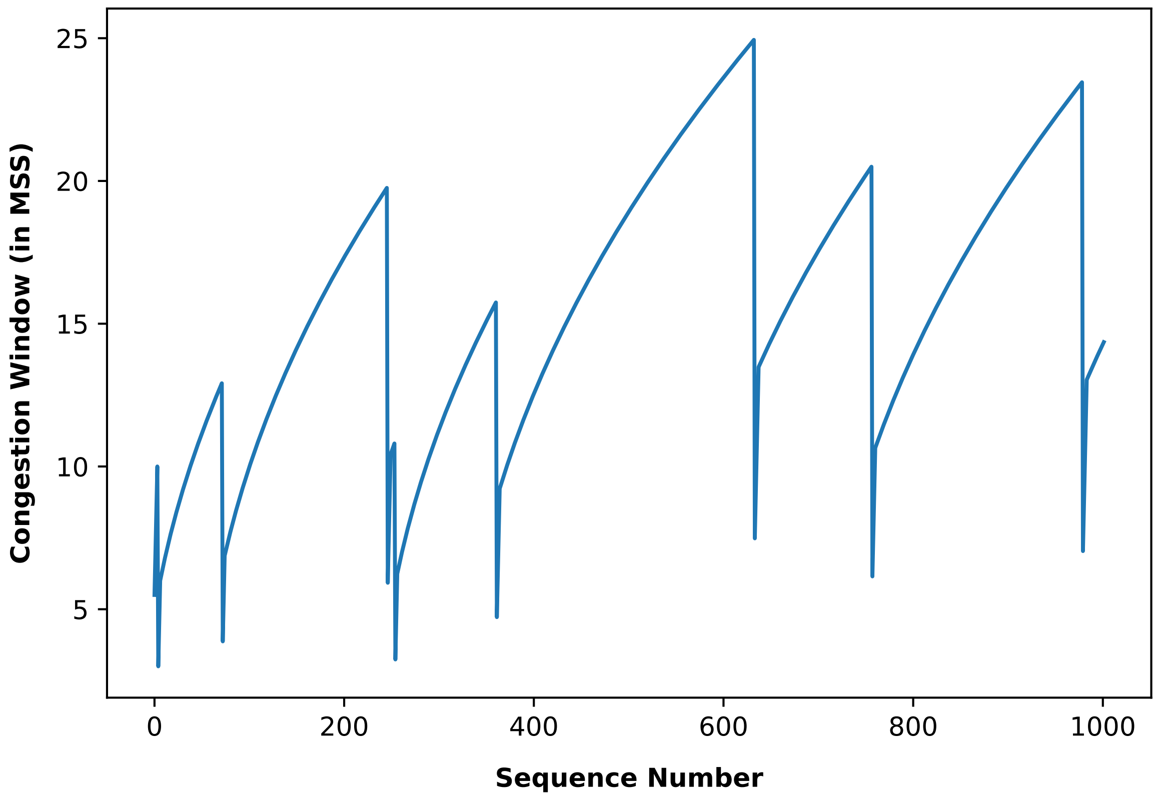

$K_i$ = **4** | $K_m$ = **1.5** | $K_n$ = **0.5** | $K_f$ = **0.1** | $P_s$ = **0.99** | $T$ = **1000**
Congestion Window (in MSS)
Sequence Number
0
10
20
30
40
50
60
70
0
200
400
600
800
1000


$K_i$ = **4** | $K_m$ = **1.5** | $K_n$ = **0.5** | $K_f$ = **0.3** | $P_s$ = **0.99** | $T$ = **1000**
Congestion Window (in MSS)
Sequence Number
0
20
40
60
80
100
120
0
200
400
600
800
1000

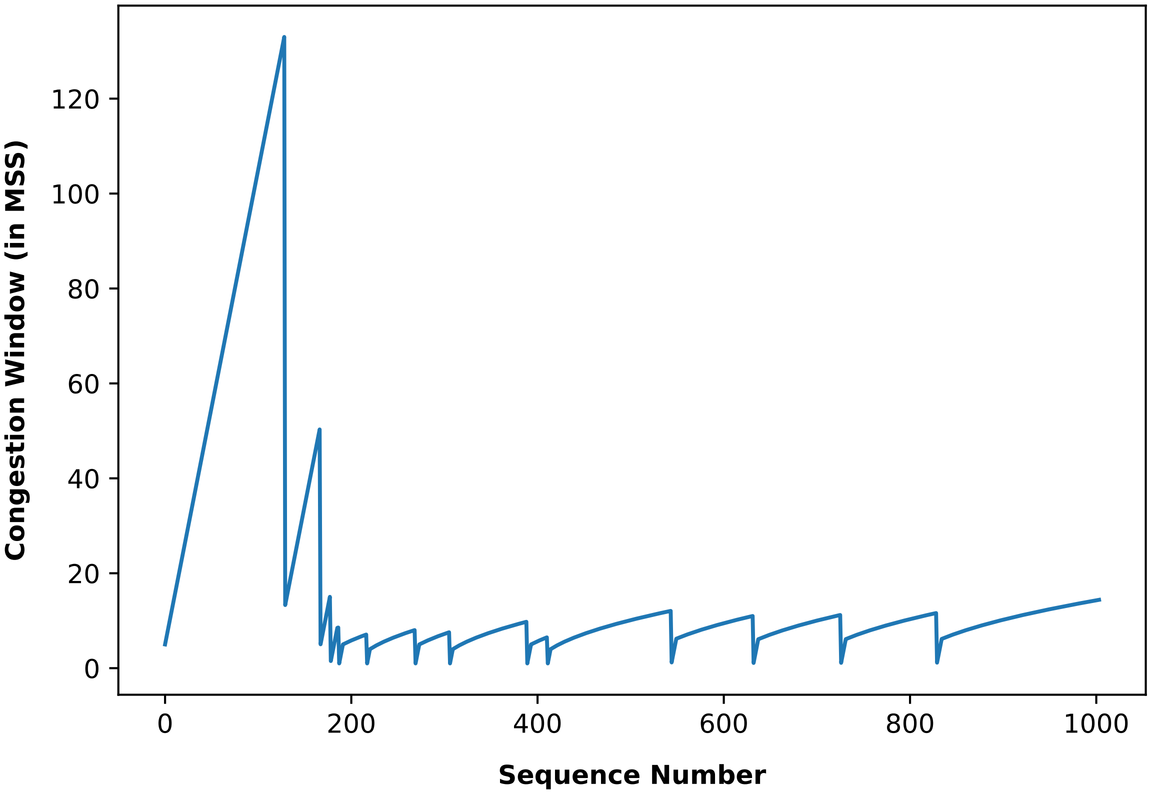
$K_i$ = **4** | $K_m$ = **1.5** | $K_n$ = **1** | $K_f$ = **0.1** | $P_s$ = **0.99** | $T$ = **1000**



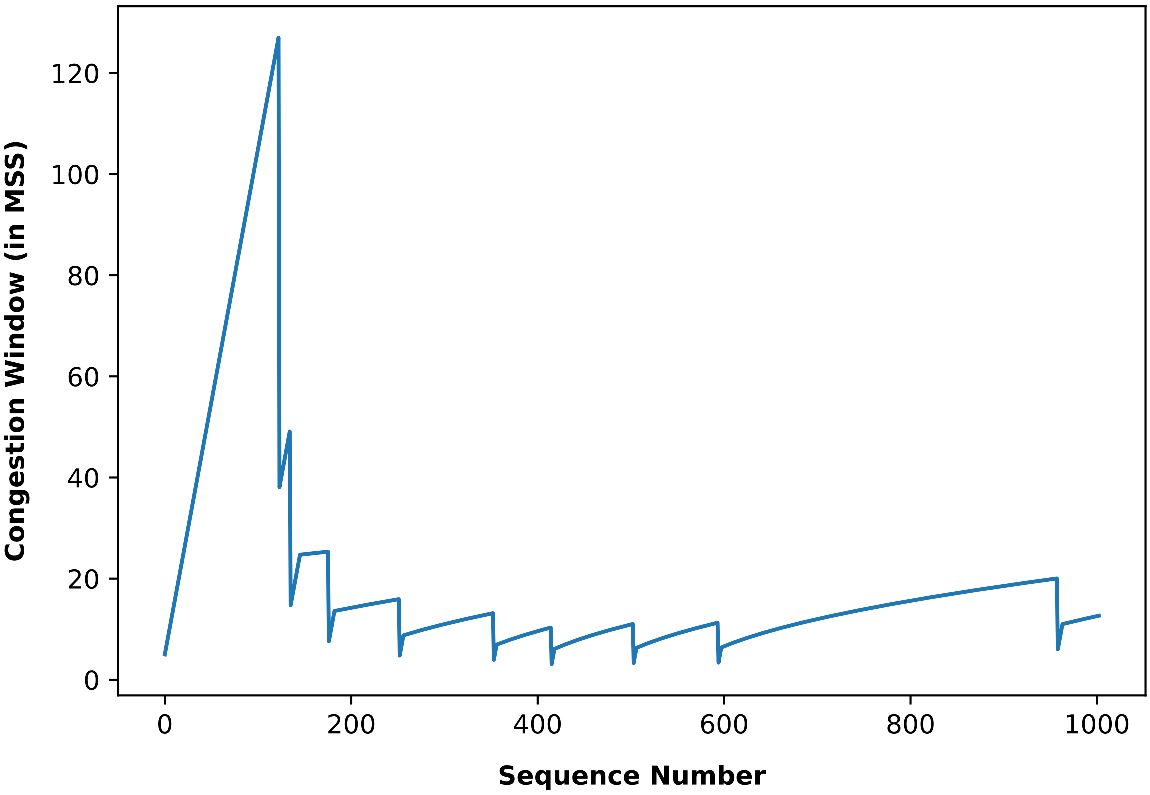

$K_i$ = **4** | $K_m$ = **1.5** | $K_n$ = **1** | $K_f$ = **0.3** | $P_s$ = **0.99** | $T$ = **1000**

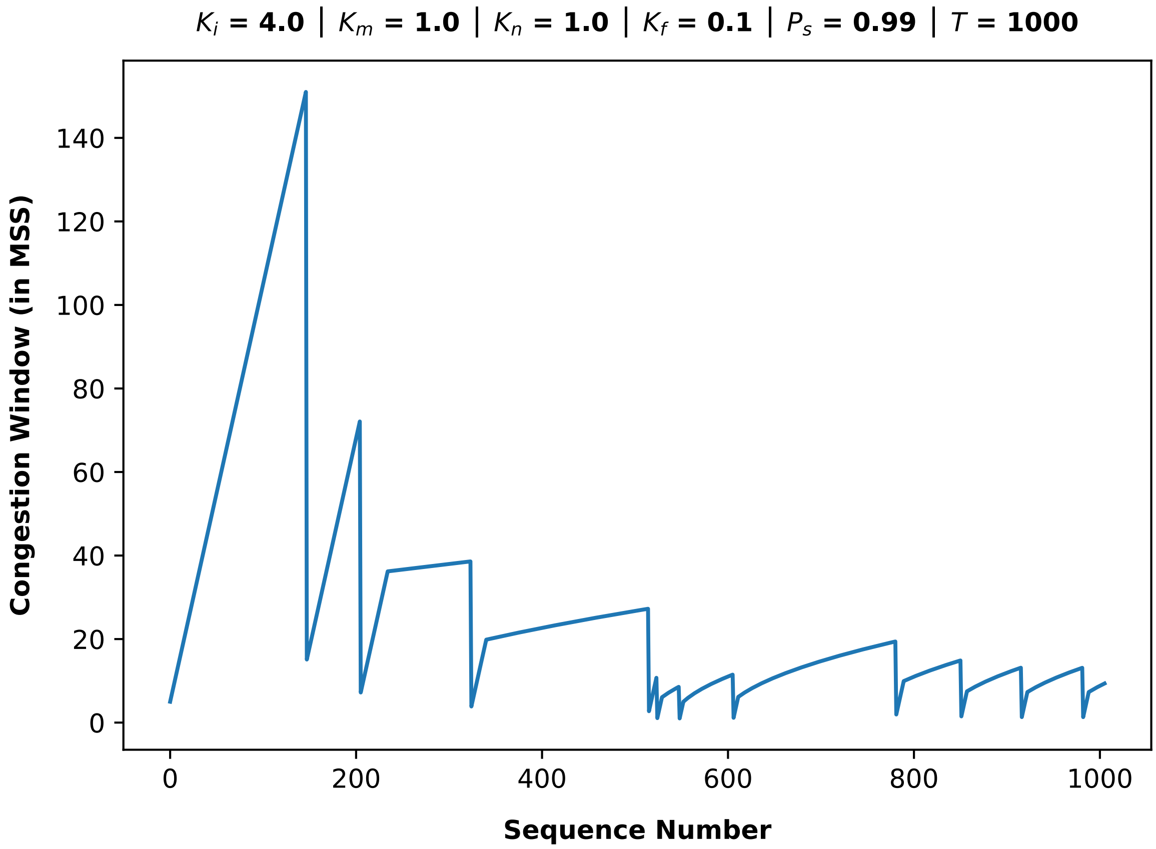


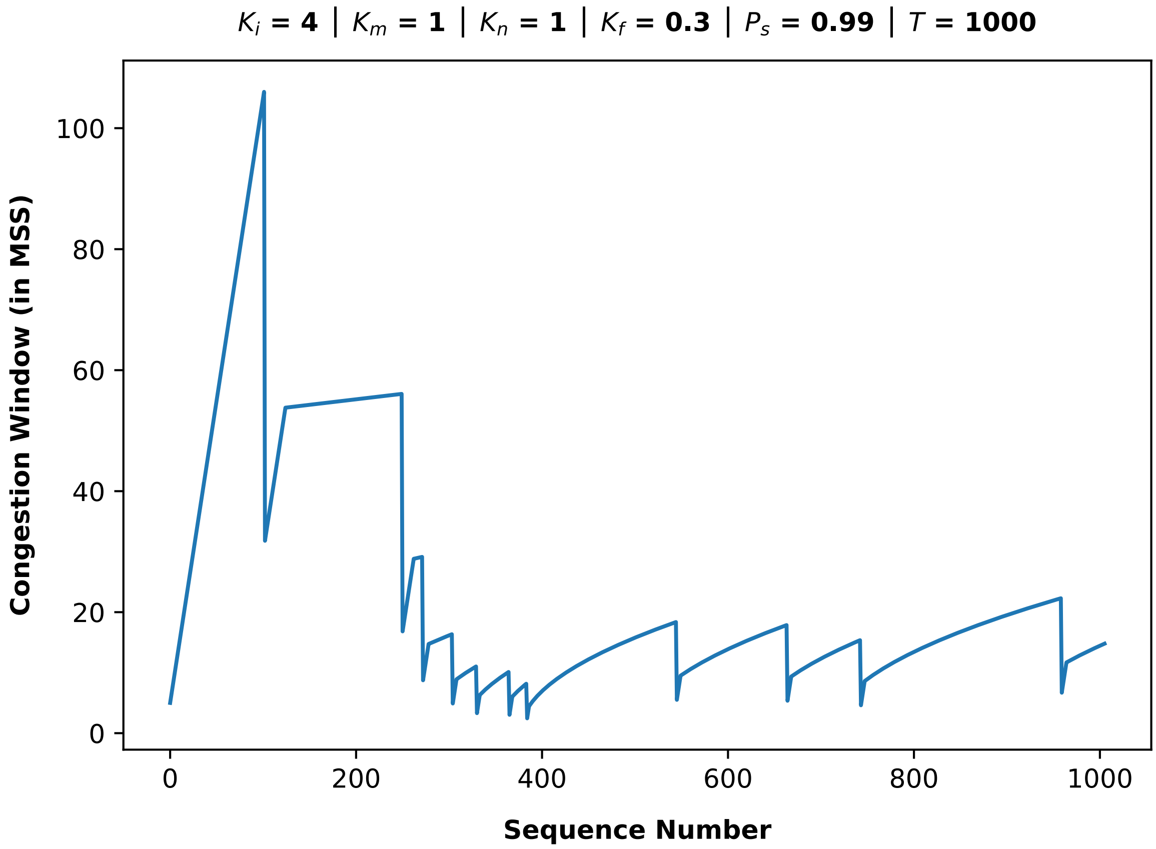
$K_i$ = **4** | $K_m$ = **1** | $K_n$ = **0.5** | $K_f$ = **0.1** | $P_s$ = **0.99** | $T$ = **1000**




$K_i$ = **4** | $K_m$ = **1** | $K_n$ = **0.5** | $K_f$ = **0.3** | $P_s$ = **0.99** | $T$ = **1000**



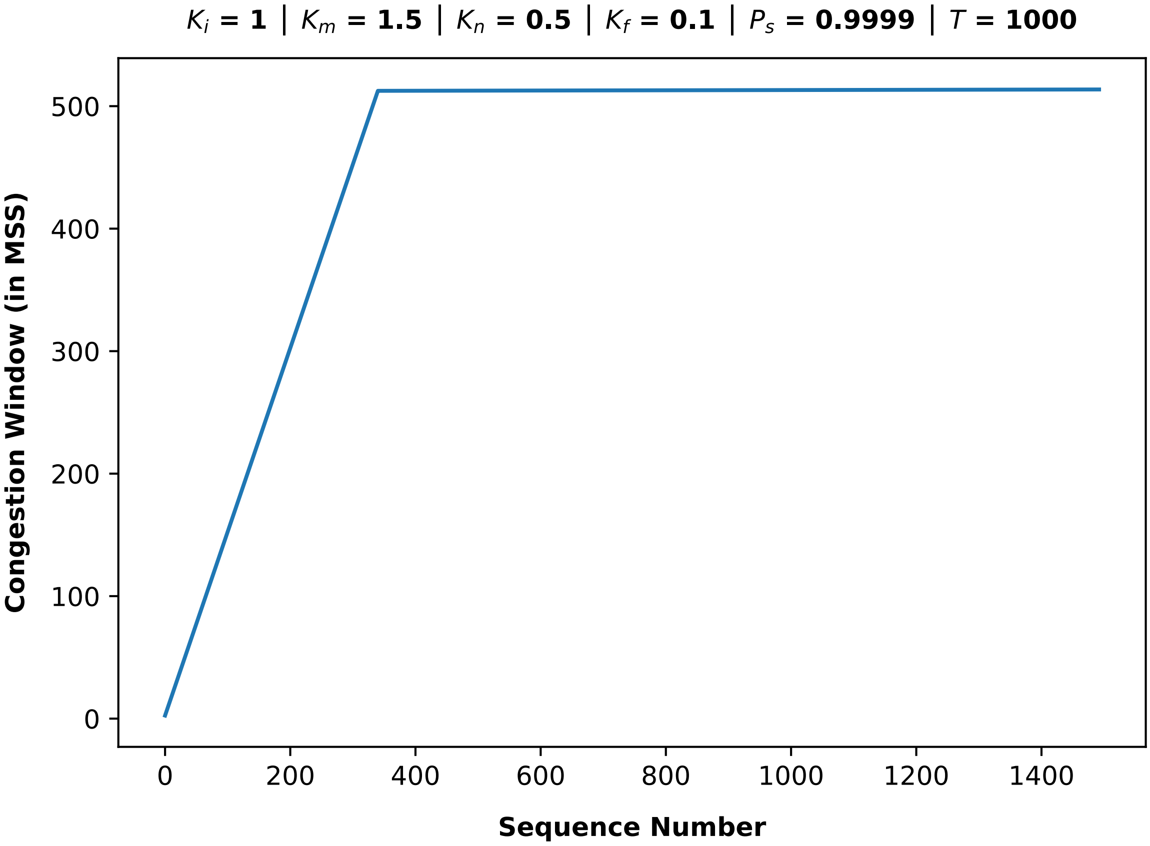

$K_i$ = 4.0 | $K_m$ = 1.0 | $K_n$ = 1.0 | $K_f$ = 0.1 | $P_s$ = 0.99 | $T$ = 1000
Congestion Window (in MSS)
0
20
40
60
80
100
120
140
0
200
400
600
800
1000
Sequence Number

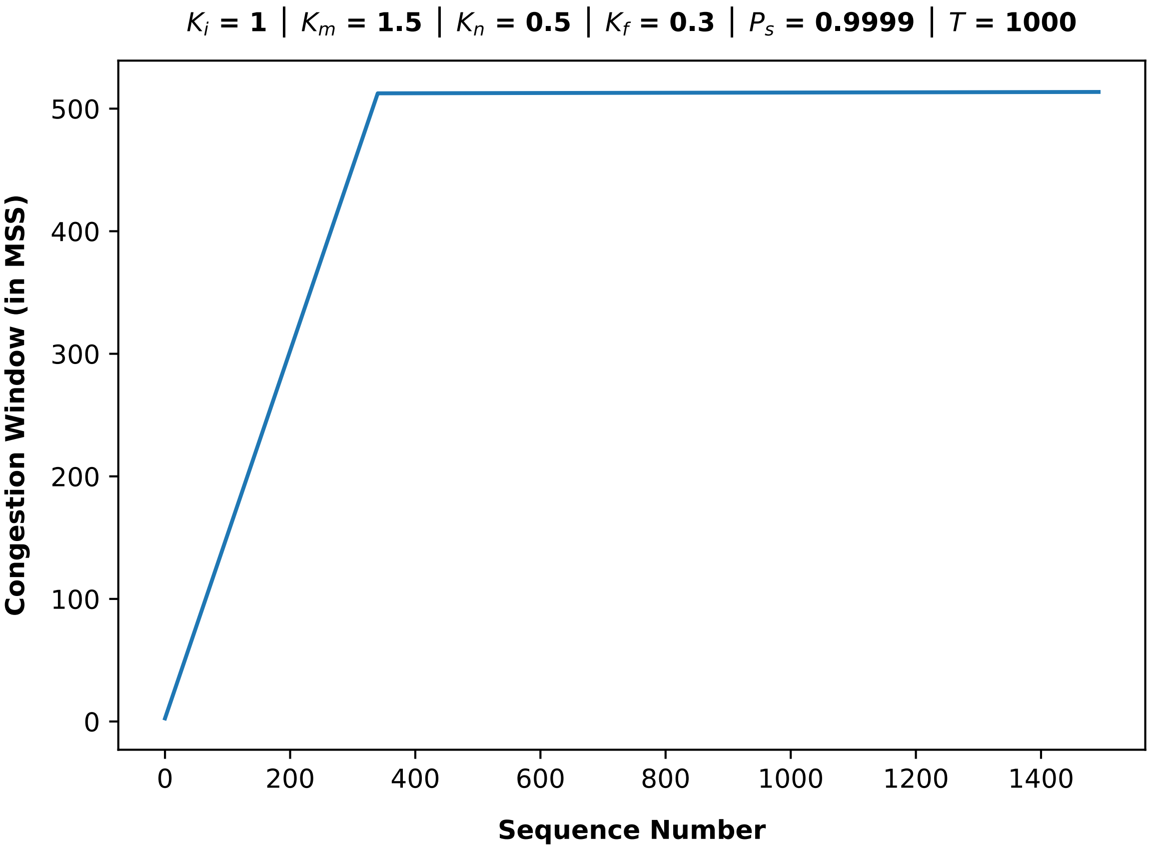

$K_i$ = 4 | $K_m$ = 1 | $K_n$ = 1 | $K_f$ = 0.3 | $P_s$ = 0.99 | $T$ = 1000
Congestion Window (in MSS)
0
20
40
60
80
100
0
200
400
600
800
1000
Sequence Number

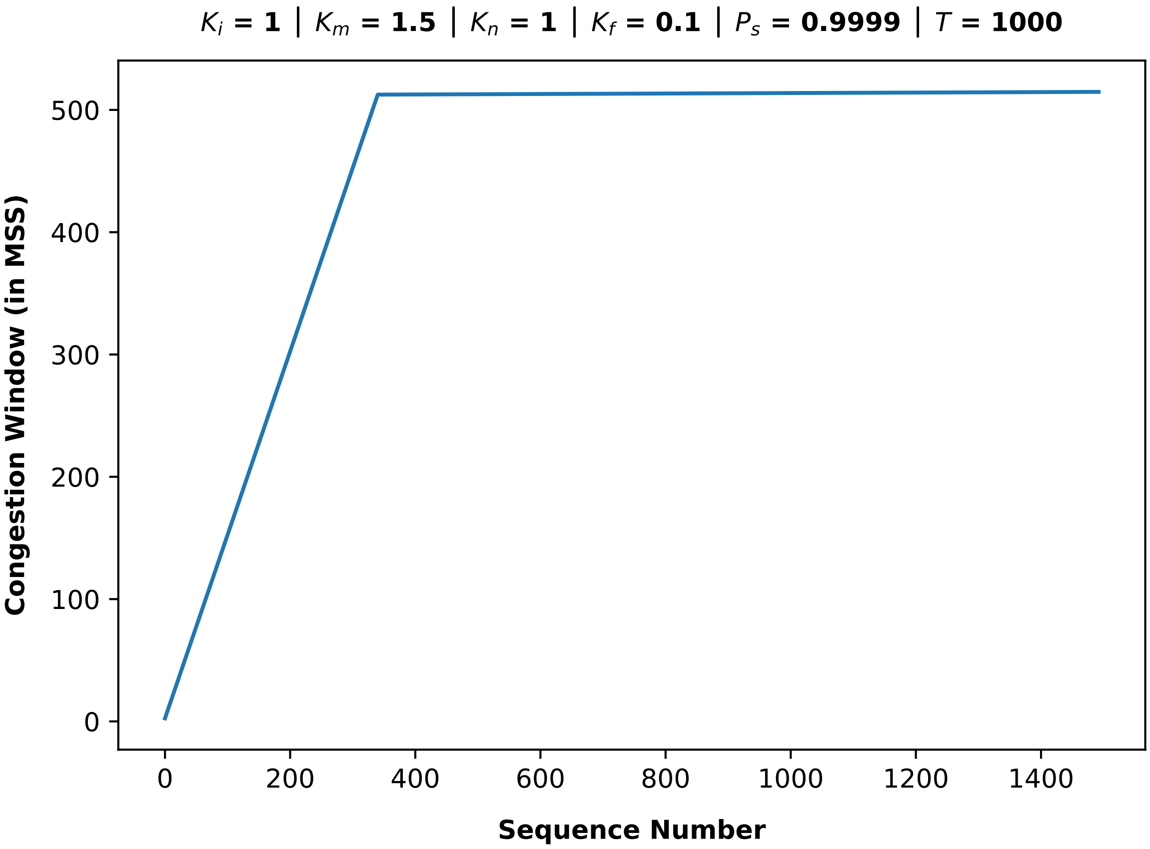
## $P_s = 0.9999$

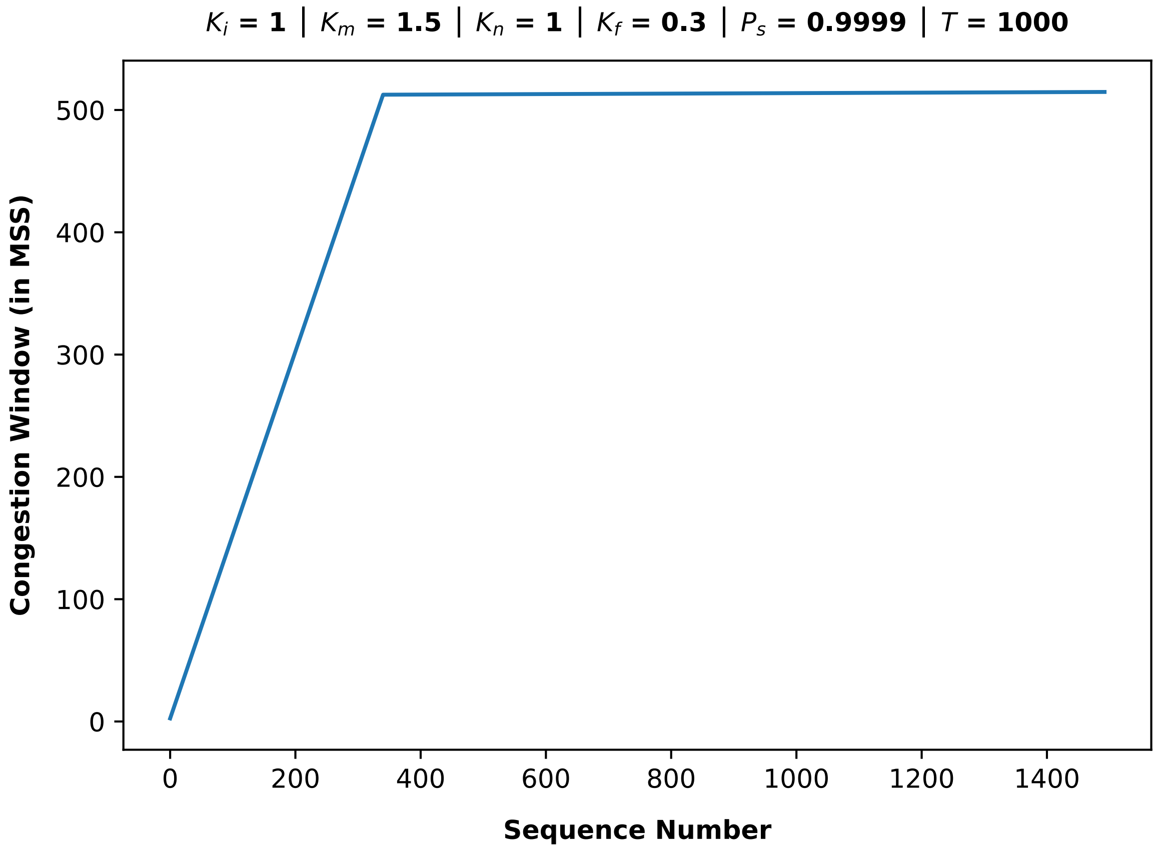
When we set the probability of a successful send to 0.9999, we get very bland graphs. For almost all the parameters, we see that the graph goes linearly (additive increase) up to the congestion threshold ($RSS/2 = 512kB$) and then becomes constant at that.

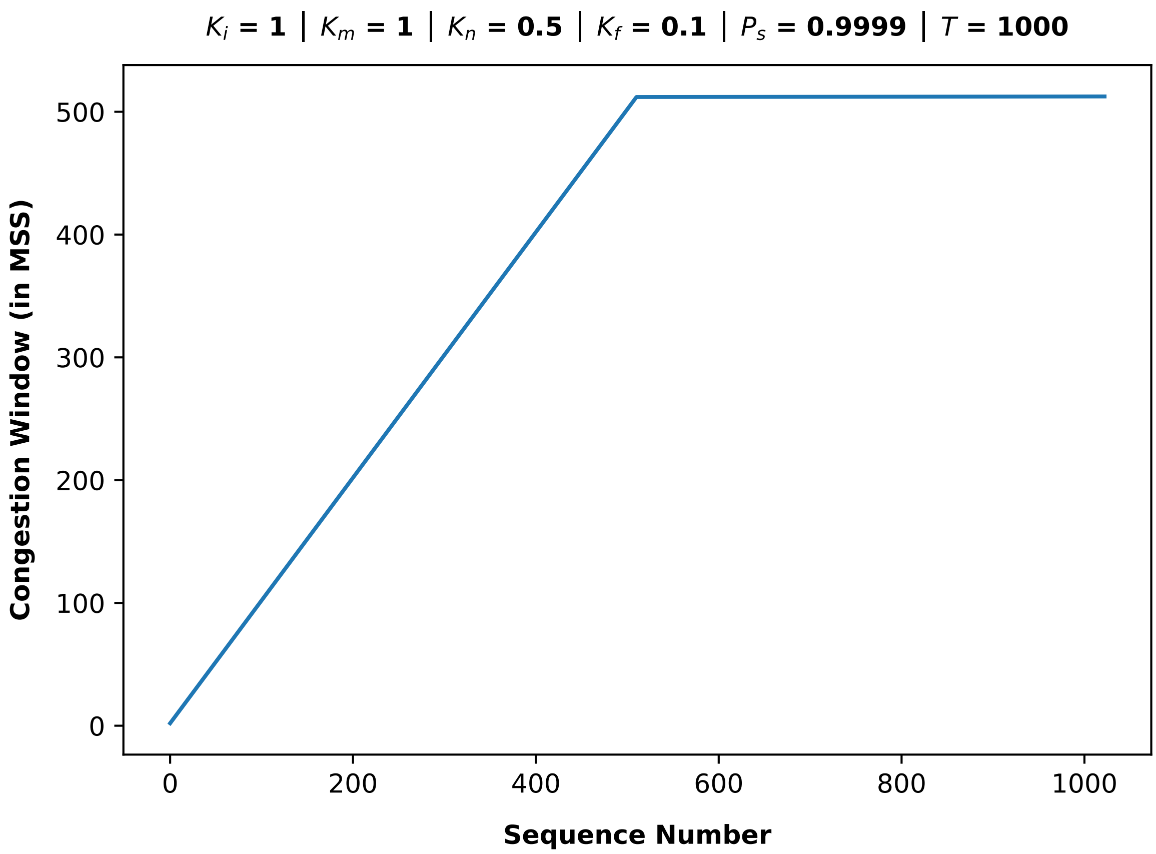
A timeout doesn't occur/rarely occurs in this case.

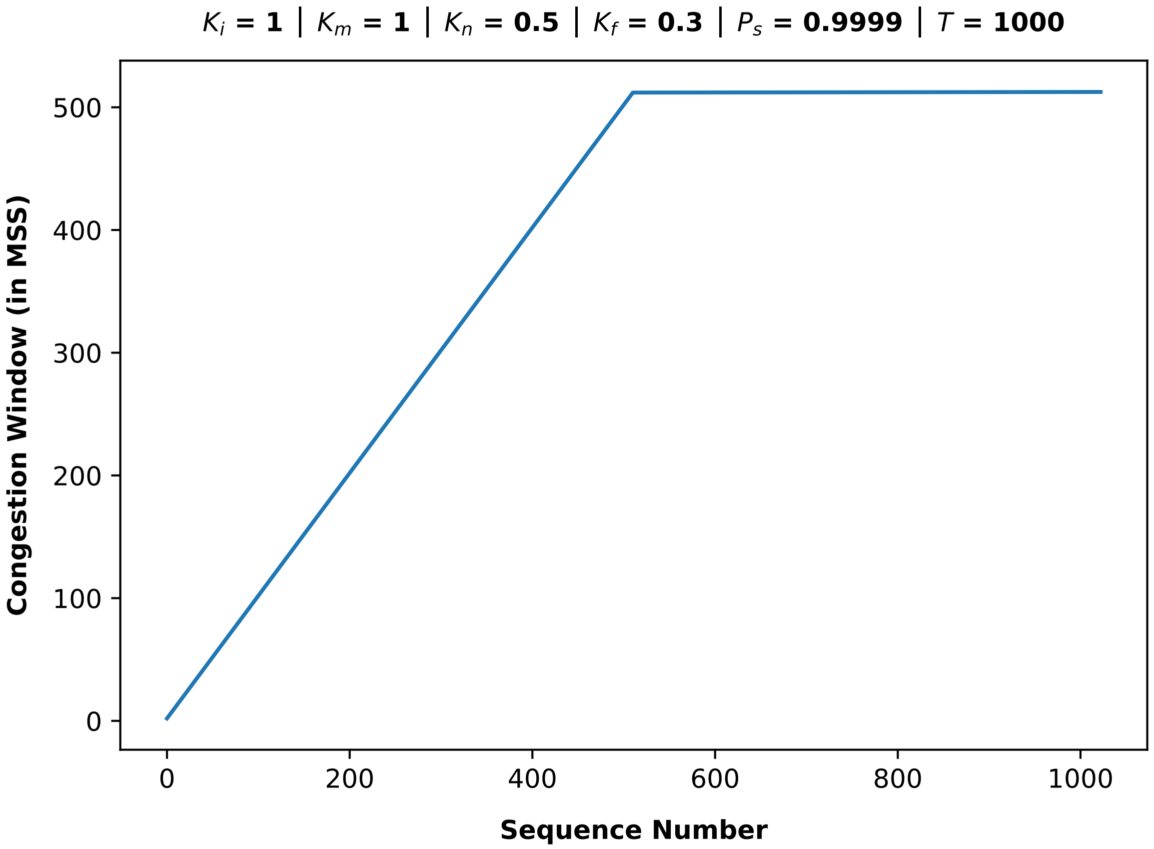


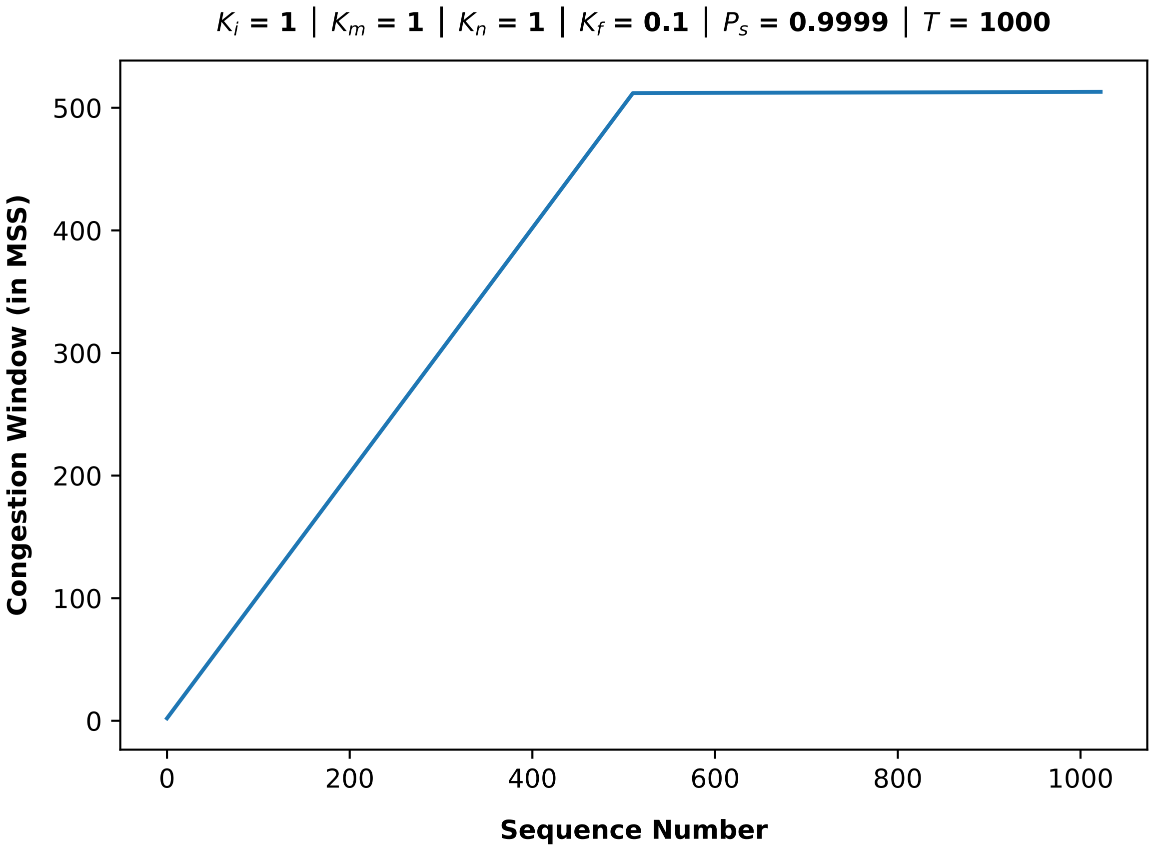

$K_i$ = 1 | $K_m$ = 1.5 | $K_n$ = 0.5 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
1400
Sequence Number

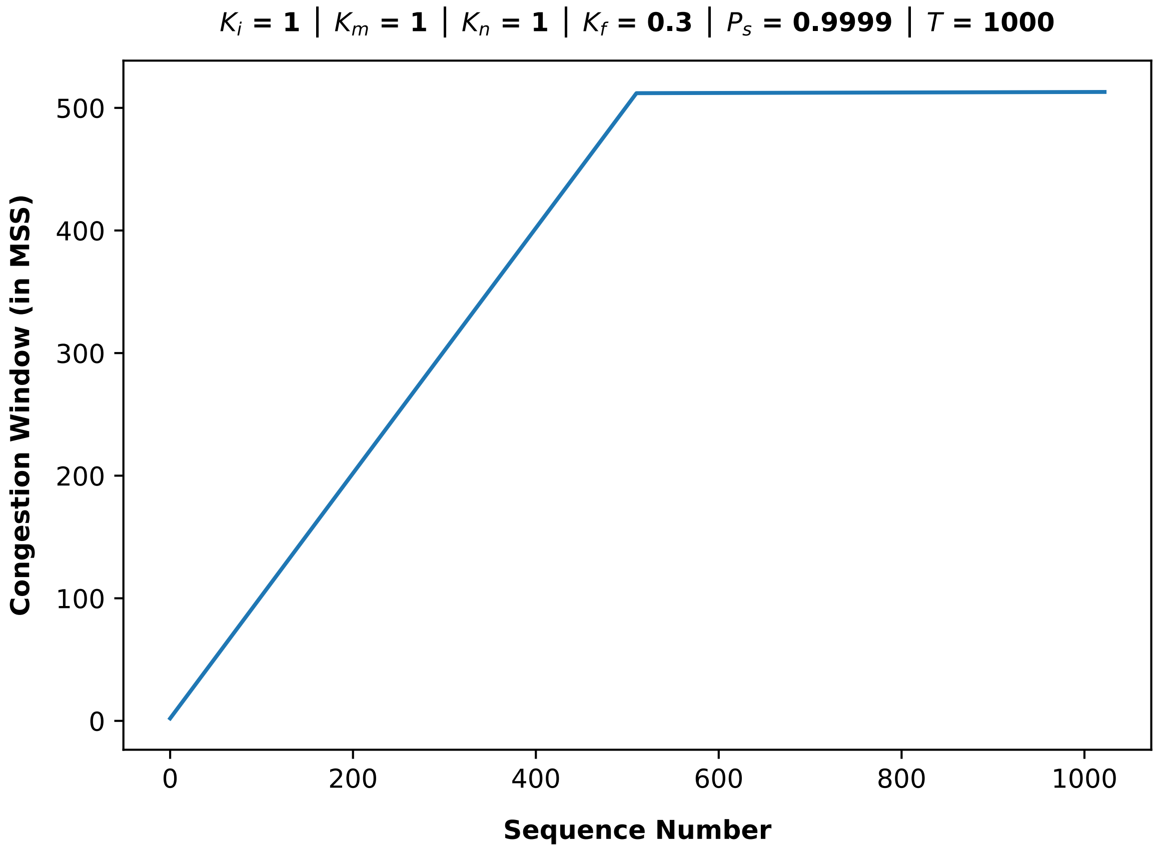

$K_i$ = 1 | $K_m$ = 1.5 | $K_n$ = 1 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
1400
Sequence Number

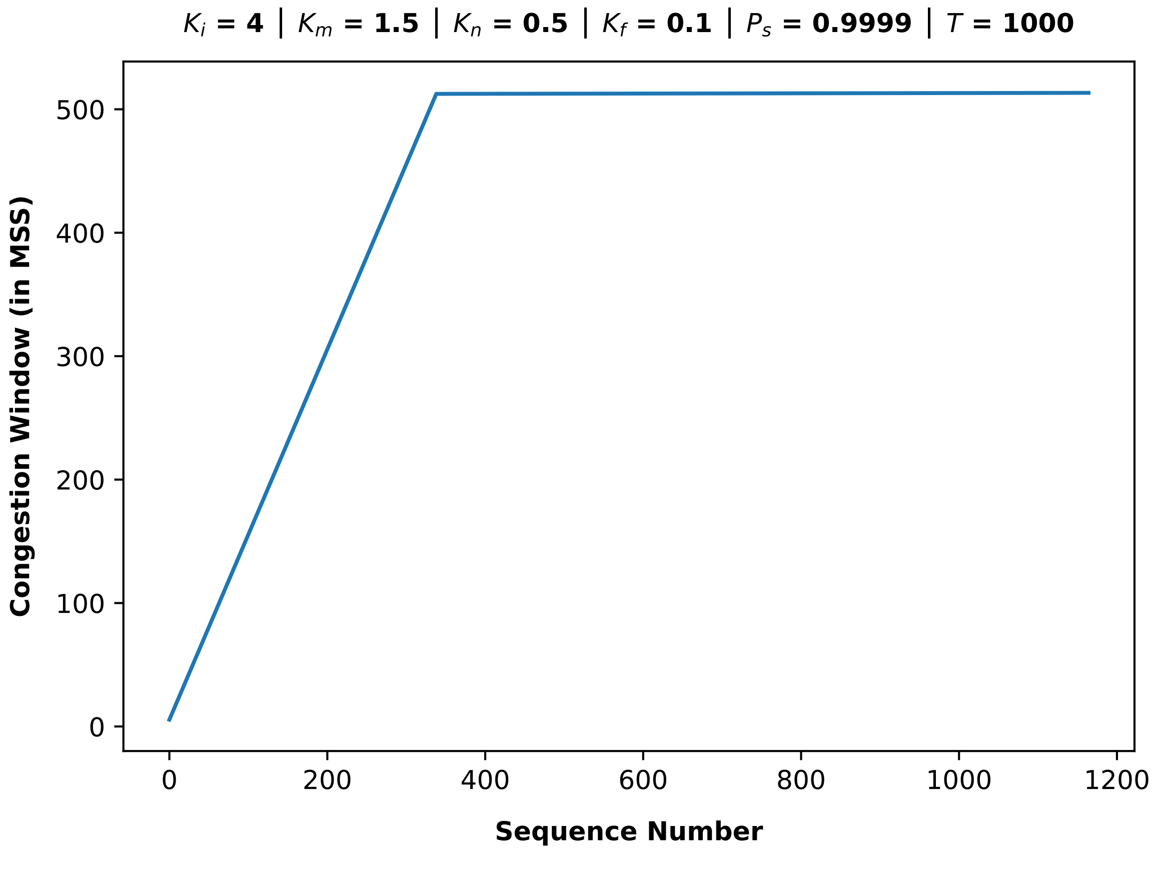

$K_i$ = 1 | $K_m$ = 1.5 | $K_n$ = 1 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
1400
Sequence Number

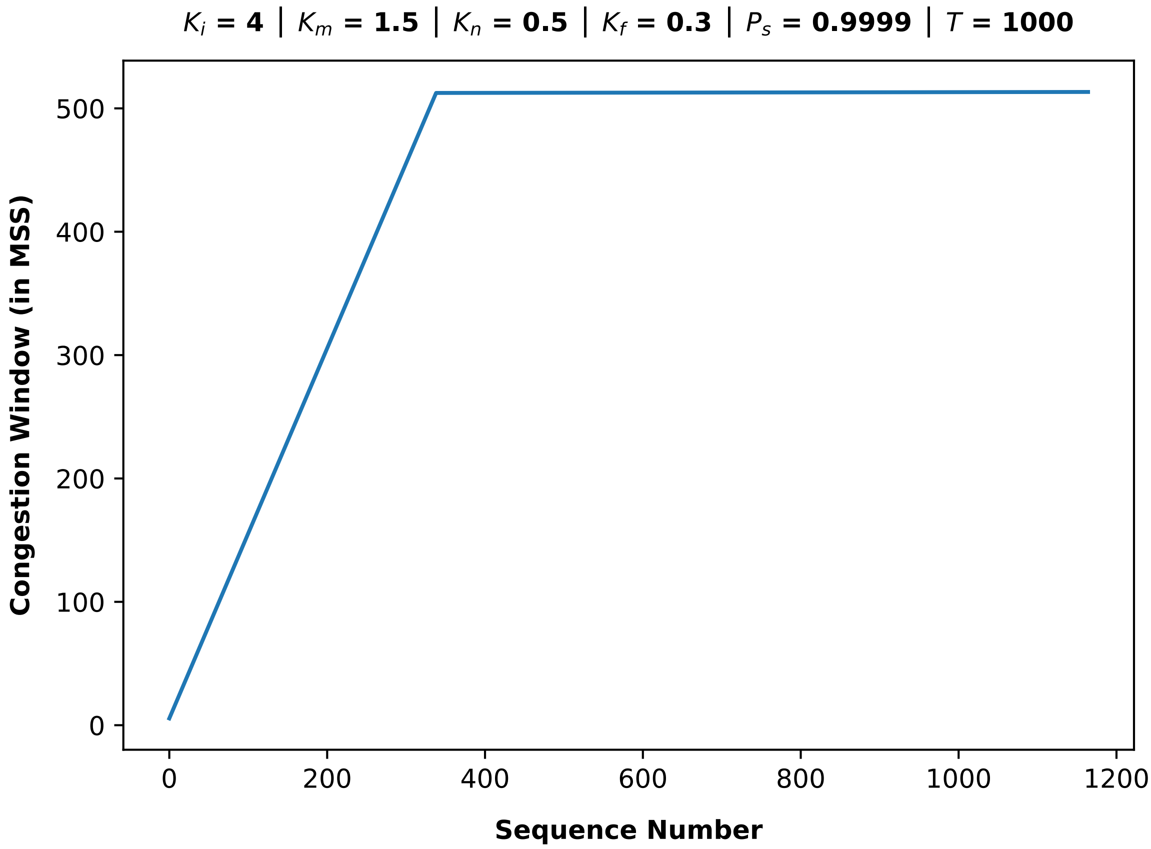

$K_i$ = 1 | $K_m$ = 1 | $K_n$ = 0.5 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
Sequence Number

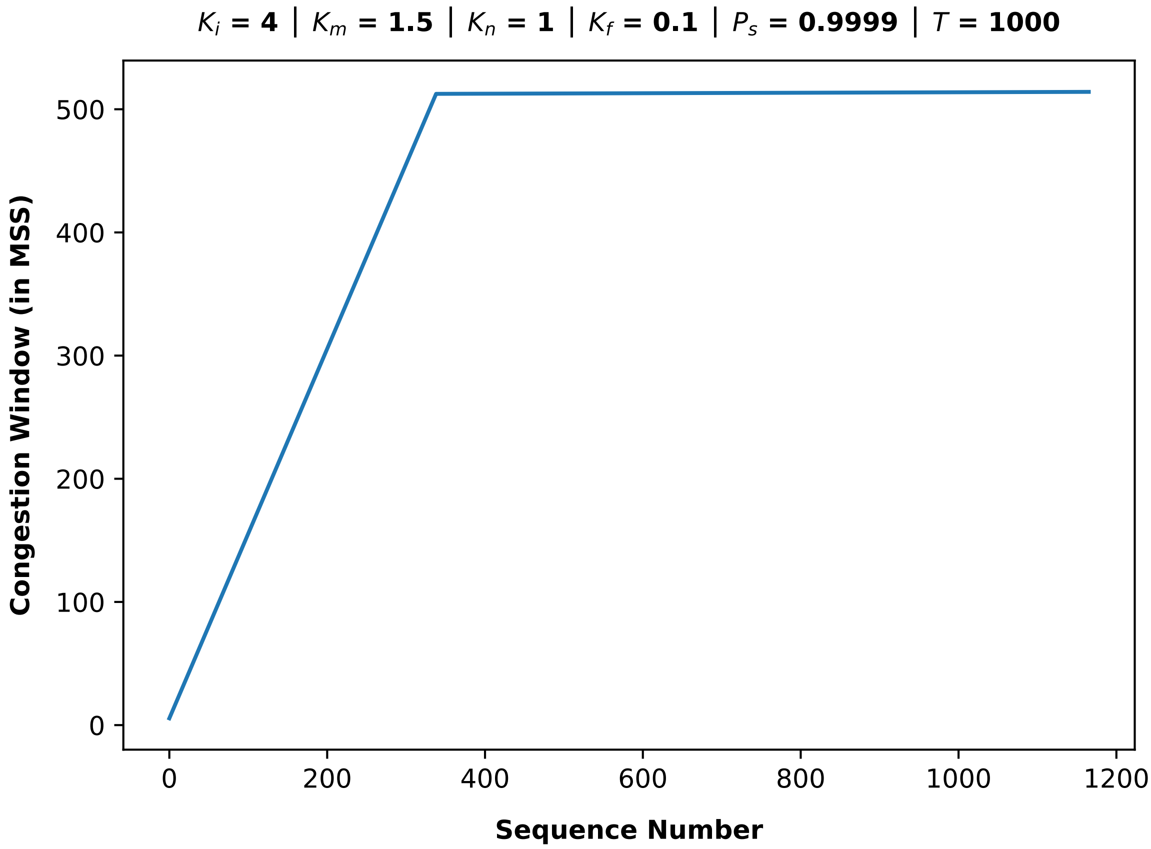

$K_i$ = 1 | $K_m$ = 1 | $K_n$ = 0.5 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
Sequence Number
0
100
200
300
400
500
0
200
400
600
800
1000

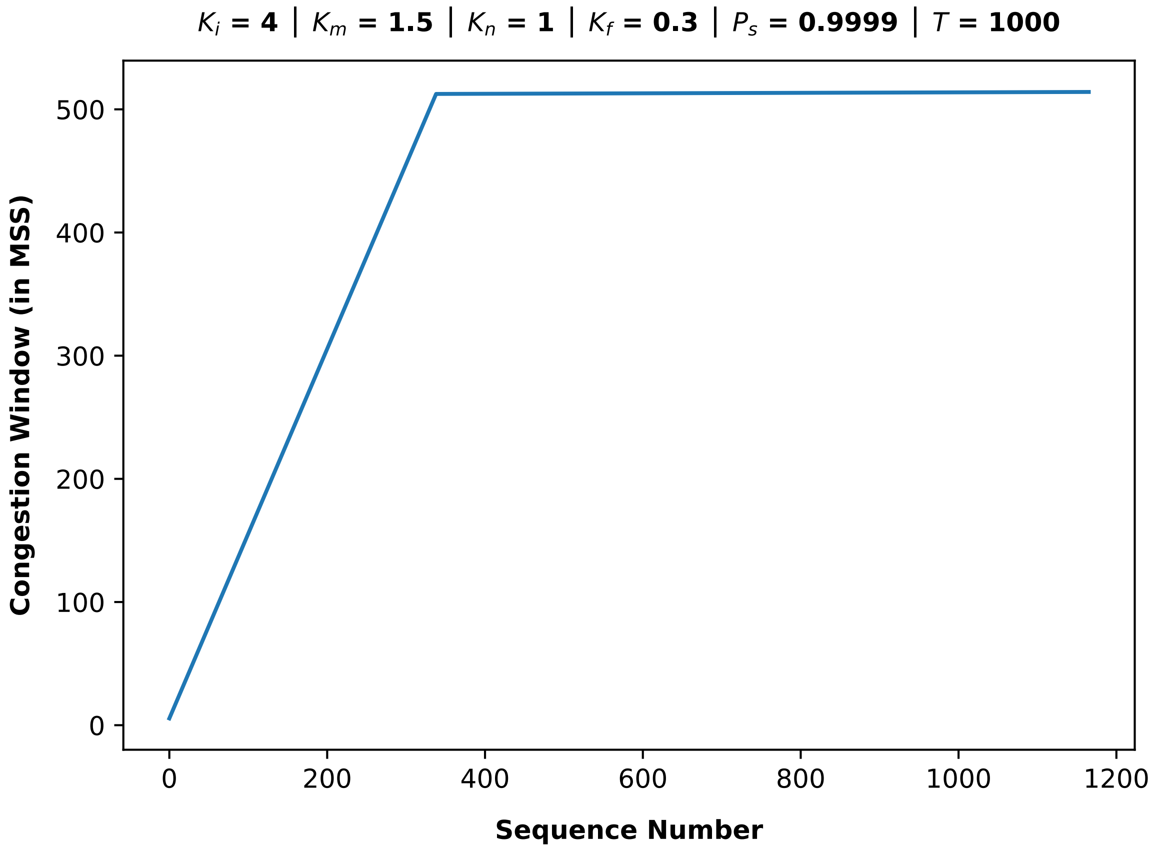

$K_i$ = 1 | $K_m$ = 1 | $K_n$ = 1 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
Sequence Number
0
100
200
300
400
500
0
200
400
600
800
1000

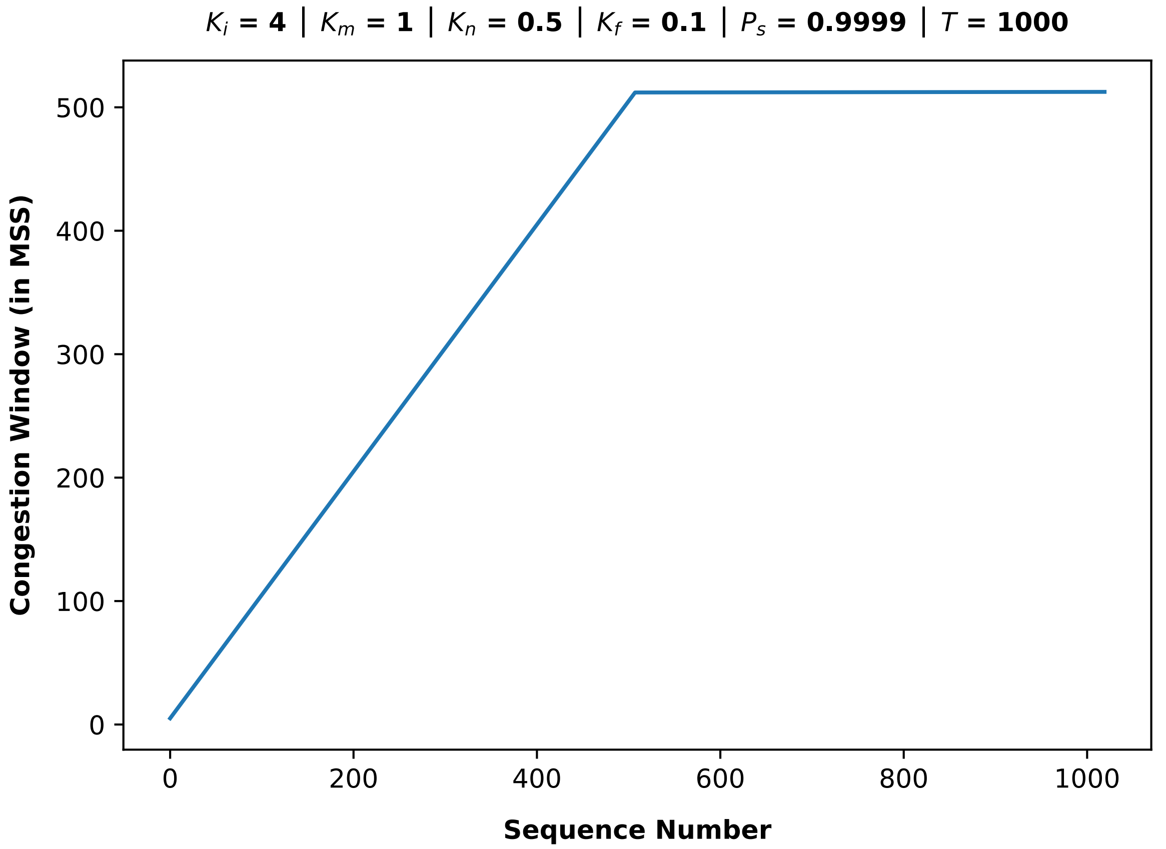

$K_i$ = 1 | $K_m$ = 1 | $K_n$ = 1 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
Sequence Number

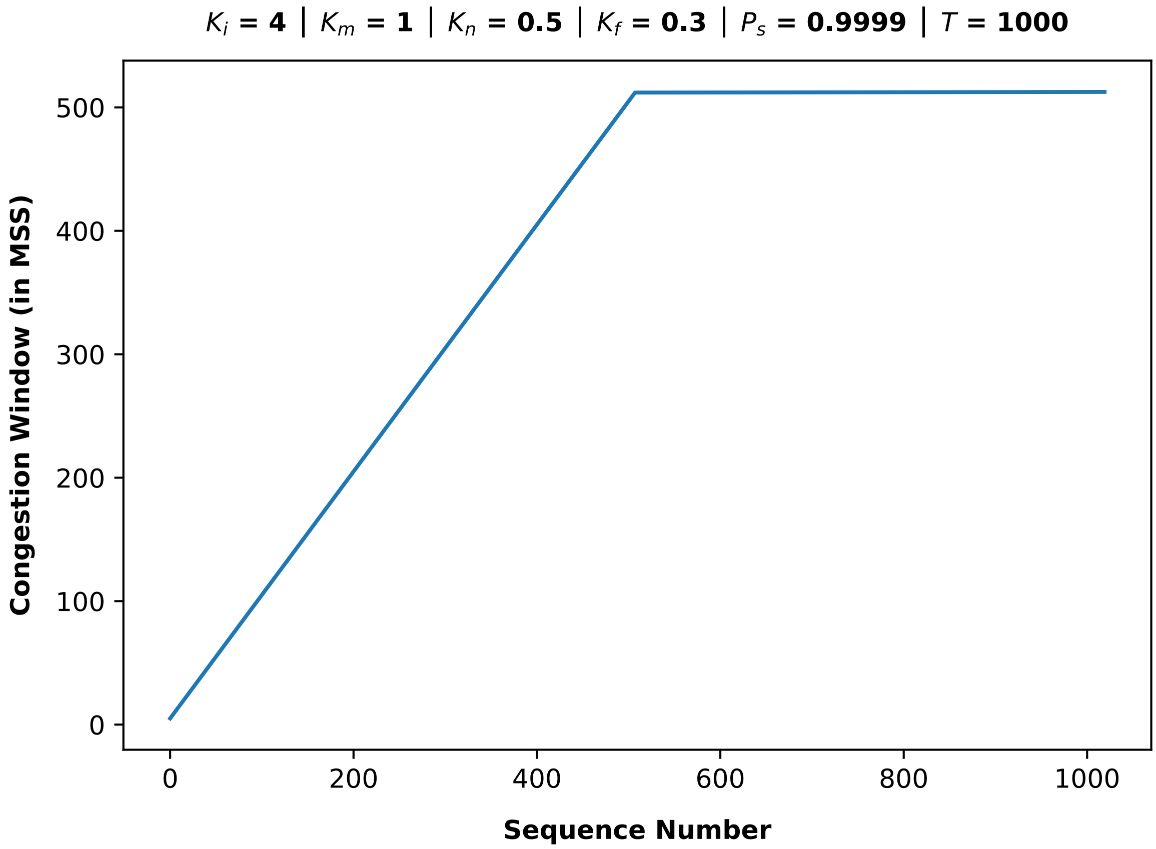

$K_i$ = 4 | $K_m$ = 1.5 | $K_n$ = 0.5 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
Sequence Number

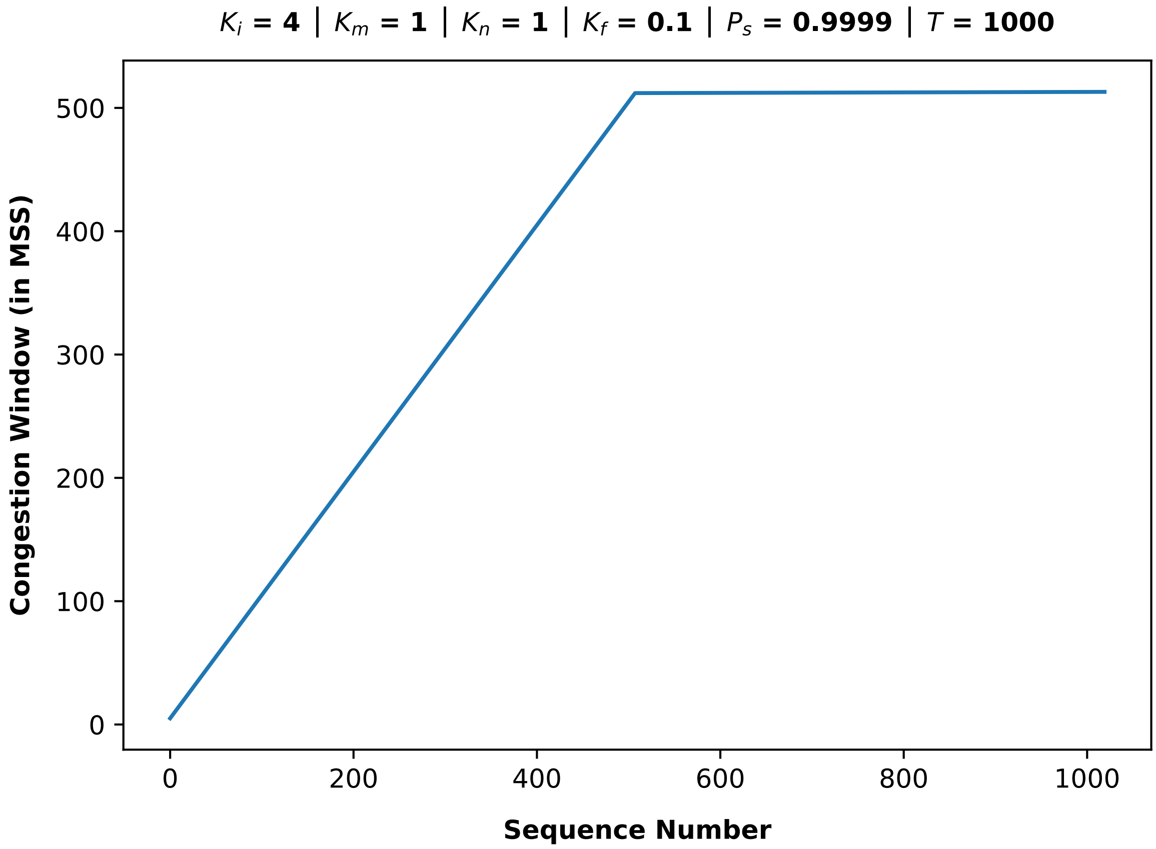

$K_i$ = 4 | $K_m$ = 1.5 | $K_n$ = 0.5 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
Sequence Number

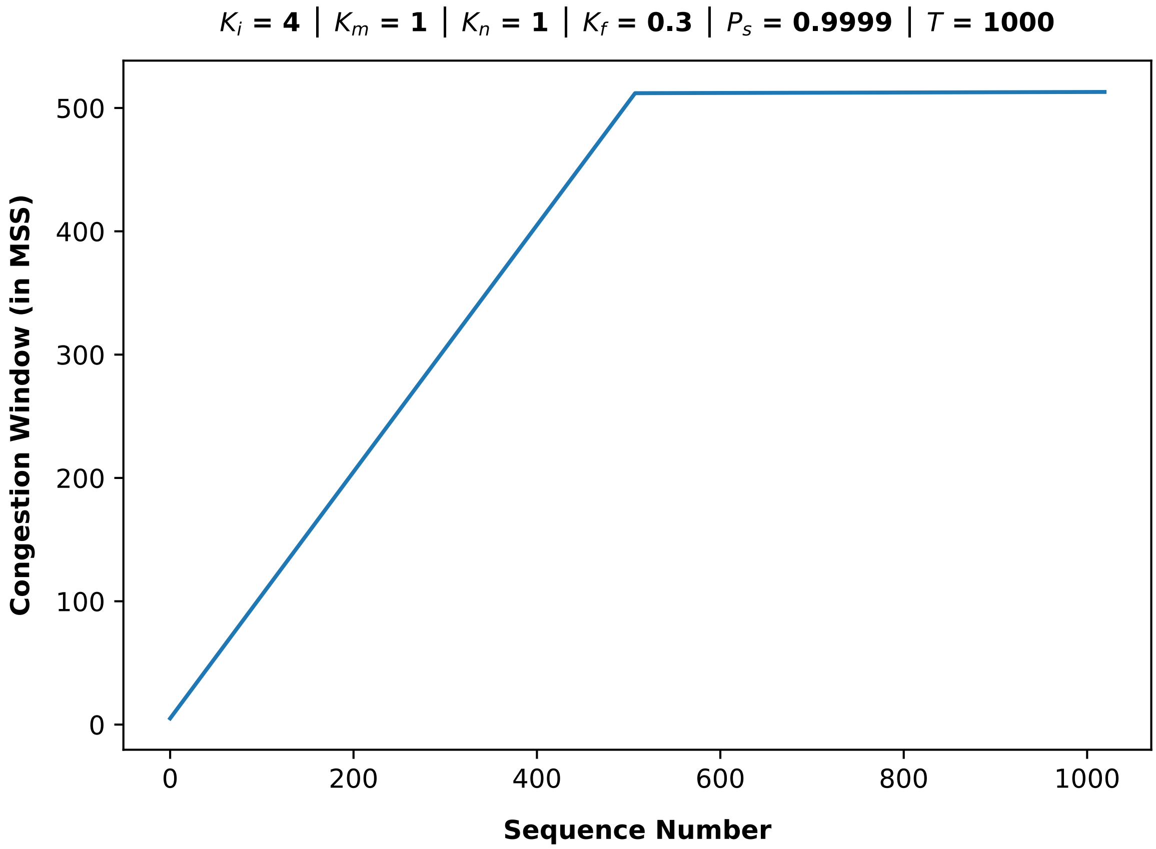

$K_i$ = 4 | $K_m$ = 1.5 | $K_n$ = 1 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
Sequence Number


$K_i$ = 4 | $K_m$ = 1.5 | $K_n$ = 1 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
1200
Sequence Number


$K_i$ = 4 | $K_m$ = 1 | $K_n$ = 0.5 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
Sequence Number


$K_i$ = 4 | $K_m$ = 1 | $K_n$ = 0.5 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
Sequence Number
0
100
200
300
400
500
0
200
400
600
800
1000


$K_i$ = 4 | $K_m$ = 1 | $K_n$ = 1 | $K_f$ = 0.1 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
Sequence Number
0
100
200
300
400
500
0
200
400
600
800
1000


$K_i$ = 4 | $K_m$ = 1 | $K_n$ = 1 | $K_f$ = 0.3 | $P_s$ = 0.9999 | $T$ = 1000
Congestion Window (in MSS)
0
100
200
300
400
500
0
200
400
600
800
1000
Sequence Number

# Conclusion

- The $CW$ values depend on all of the parameters but they are also heavily dependent on the probability of successful `ACK`.
- The highest $CW$ matters in determining the shape of the graph.
- A higher $K_f$, $K_m$ and $K_n$ value generally give higher $CW$ values.
- An extra parameter can also be tested, i.e, the factor by which the congestion threshold was decreased.
- The formulae suggest an AIMD (Additive Increase Multiplicative Decrease) algorithm.
- The terms *exponential* and *linear* are kind of misnomers as during the phase where $CW$ is less than the threshold, it increases rapidly but linearly whereas during the later phase when the $CW$ is higher than the threshold, it increases in a logarithmic manner.