CS 421- PROGRAMMING ASSIGNMENT 2

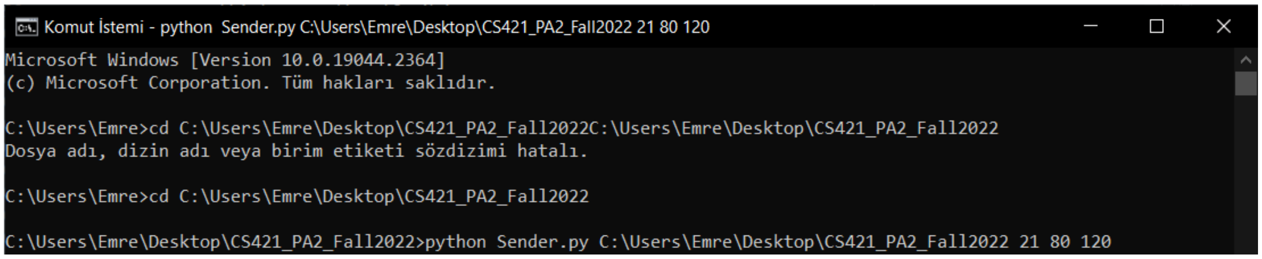
Emre Can Şen-21902516

INTRODUCTION

In this programming assignment, a client sender program with selective repeat was implemented. Designed client program is able to send lost packages until an acknowledgement message is returned with given port, window size, timeout and file path inputs. Then the program's throughput was tested with differing loss probabilities and window sizes.

WORK AND RESULTS

The codes were run with windows command windows with given format in the assignment. The sender code gives an error at the end but since the error does not affect the file transmission, it was left as it is. Examples of working client and sender command windows can be seen below:



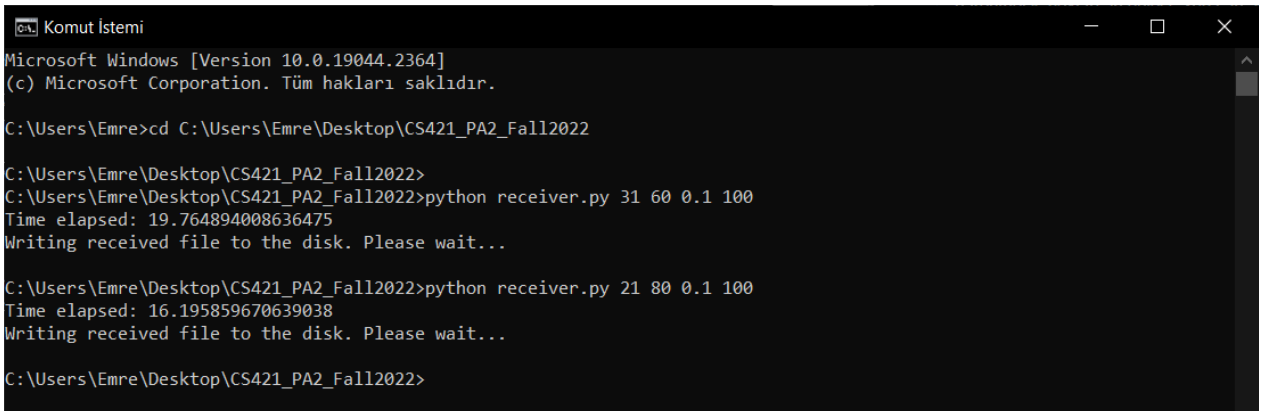
```
Komut İstemi - python  Sender.py C:\Users\Emre\Desktop\CS421_PA2_Fall2022 21 80 120
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Emre>cd C:\Users\Emre\Desktop\CS421_PA2_Fall2022C:\Users\Emre\Desktop\CS421_PA2_Fall2022
Dosya adı, dizin adı veya birim etiketi sözdizimi hatalı.

C:\Users\Emre>cd C:\Users\Emre\Desktop\CS421_PA2_Fall2022

C:\Users\Emre\Desktop\CS421_PA2_Fall2022>python Sender.py C:\Users\Emre\Desktop\CS421_PA2_Fall2022 21 80 120
```

Figure 1- Sender Command Window



```
Komut İstemi
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Emre>cd C:\Users\Emre\Desktop\CS421_PA2_Fall2022

C:\Users\Emre\Desktop\CS421_PA2_Fall2022>
C:\Users\Emre\Desktop\CS421_PA2_Fall2022>python receiver.py 31 60 0.1 100
Time elapsed: 19.764894008636475
Writing received file to the disk. Please wait...

C:\Users\Emre\Desktop\CS421_PA2_Fall2022>python receiver.py 21 80 0.1 100
Time elapsed: 16.195859670639038
Writing received file to the disk. Please wait...

C:\Users\Emre\Desktop\CS421_PA2_Fall2022>
```

Figure 2- Receiver Command Window

Then each of the file transfer times were obtained for the given values in the assignment:

p=0.0, 17.599831581115723 seconds => 1.04 Mb/s (Throughput)

p=0.1, 29.643325090408325 seconds => 617 Kb/s

p=0.2, 42.39787530899048 seconds => 432 Kb/s

p=0.3, 57.49822235107422 seconds => 318 Kb/s

p=0.4, 71.56325674057007 seconds => 256 Kb/s p=0.5, 85.5674638748169 seconds => 214 Kb/s

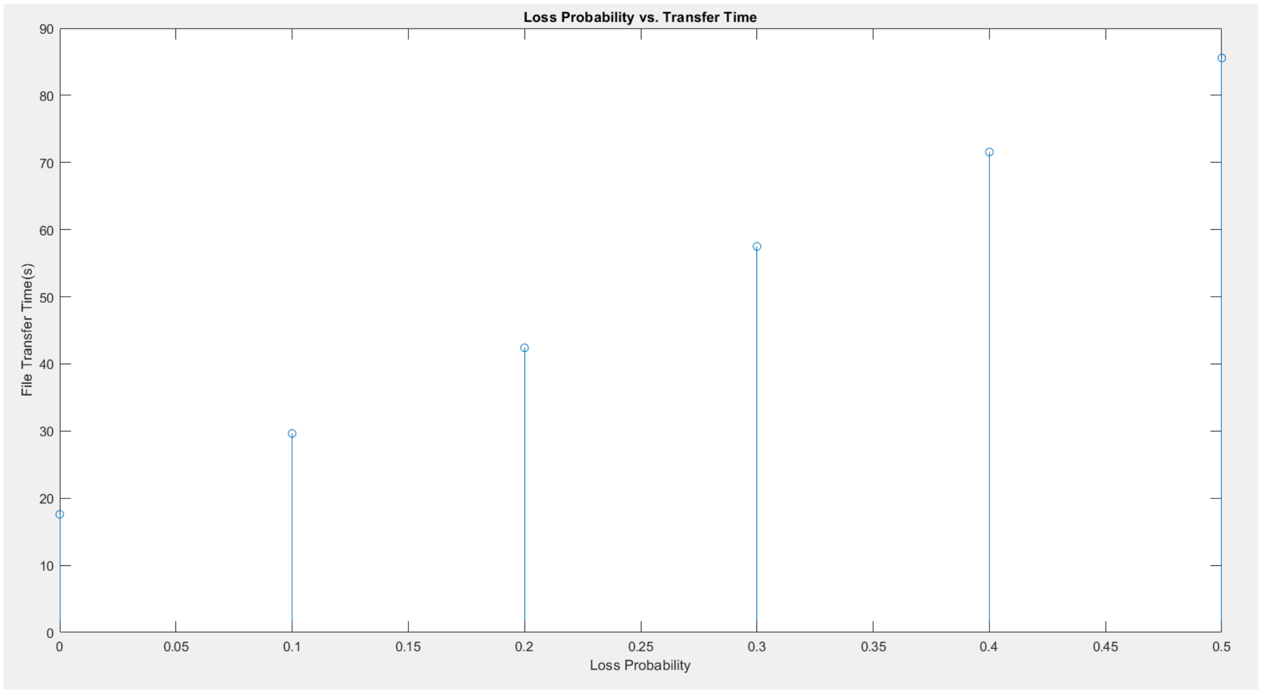


Figure 3- Loss Probability vs. Transfer Time

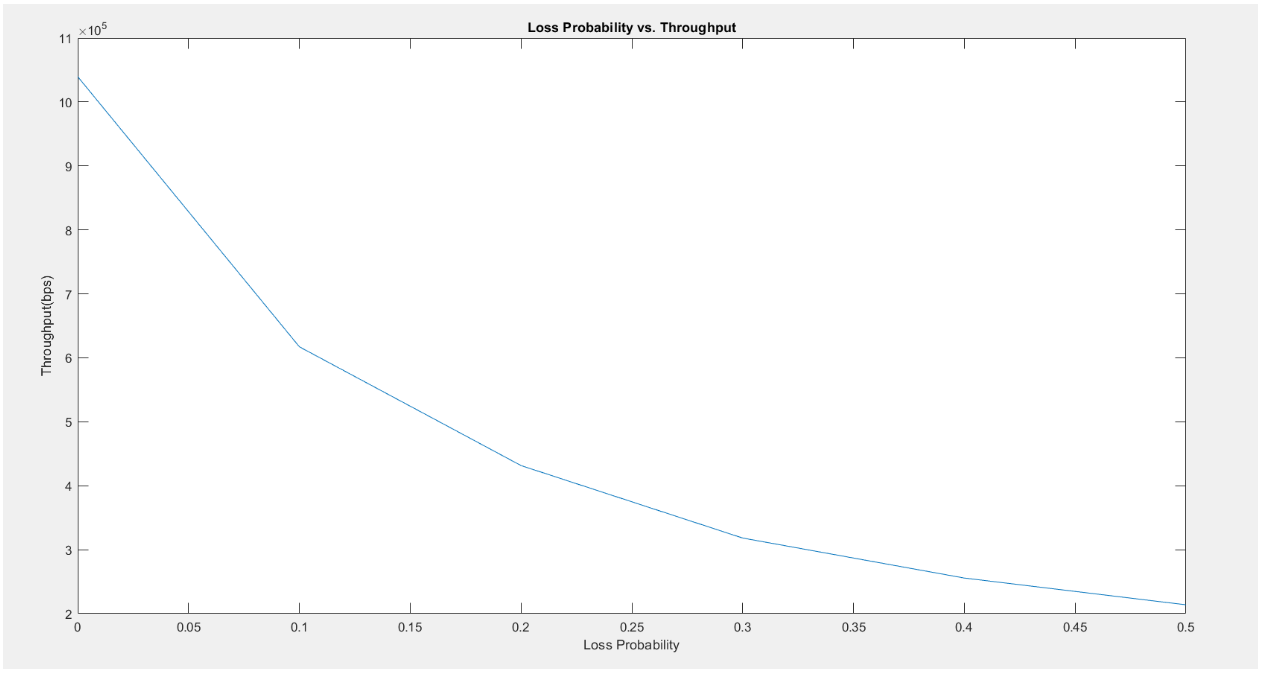


Figure 4- Loss Probability vs. Throughput

These results are pretty self-explanatory, as the chance of packet drop increases, so does the amount of retransmissions. The parameter p determines how likely a package is to be retransmitted. So when p increases, average transmission amount of a packet increases. This fact can be seen at figure 3 as the transmission times increase linearly. Then to find the throughput, file size is divided by these transmission times to get the figure 4.

Then throughput for different window sizes were determined:

N=20, 31.873268604278564 seconds => 574 Kb/s (Throughput)

N=40, 21.850029230117798 seconds => 837 Kb/s

N=60, 18.381356716156006 seconds => 995 Kb/s

N=80, 16.195859670639038 seconds => 1.13 Mb/s

N=100, 11.94794511795044 seconds => 1.53 Mb/s

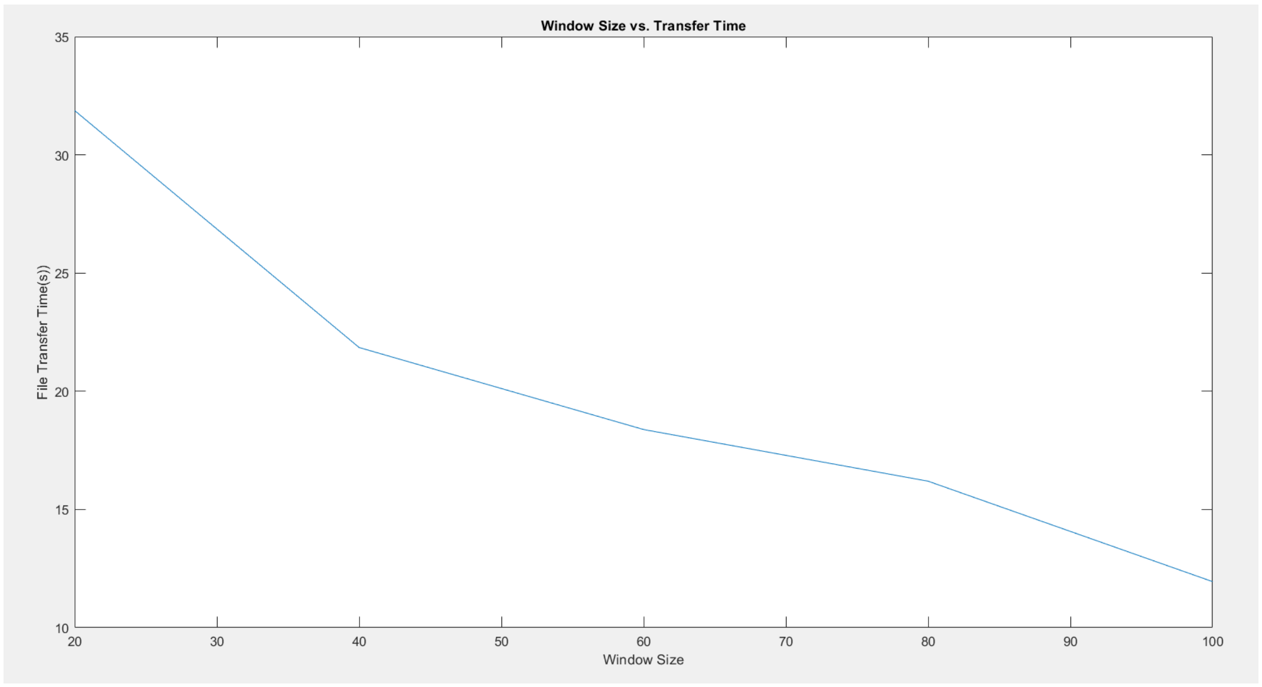


Figure 5- Window Size vs. Transfer Time

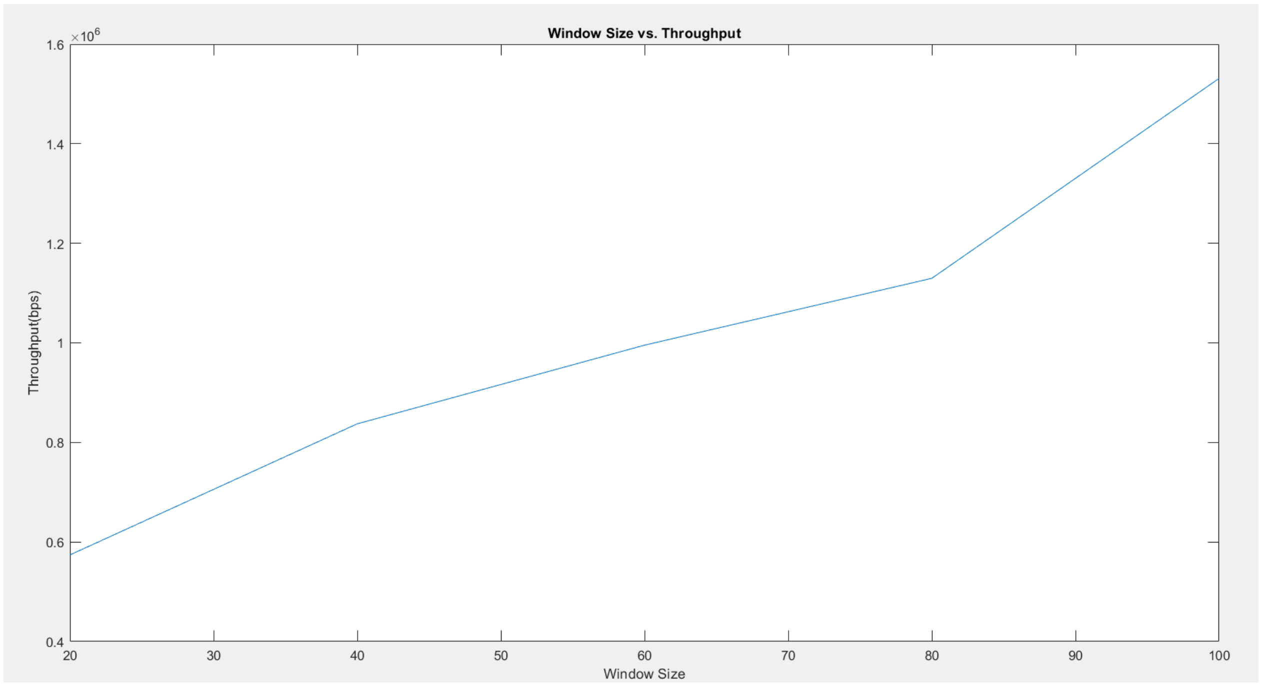


Figure 6-Window Size vs. Throughput

Higher window size means that client will send higher number of packets back to back and then retransmit timeouts. When waiting for timeout to occur, that time is wasted. So when higher number of window size is used, less time will be wasted waiting, consequently increasing the throughput.

CONCLUSION

In this programming assignment, a selective repeat client was implemented. It was observed that as the packet loss probability increased linearly, so did the transmission time. Also increase in window size increased the throughput, since the time waiting for timeout to occur was decreased.