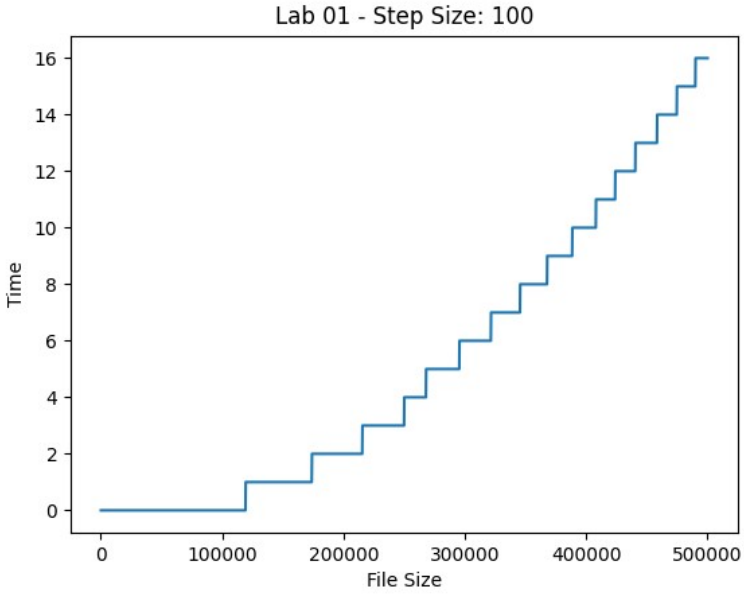
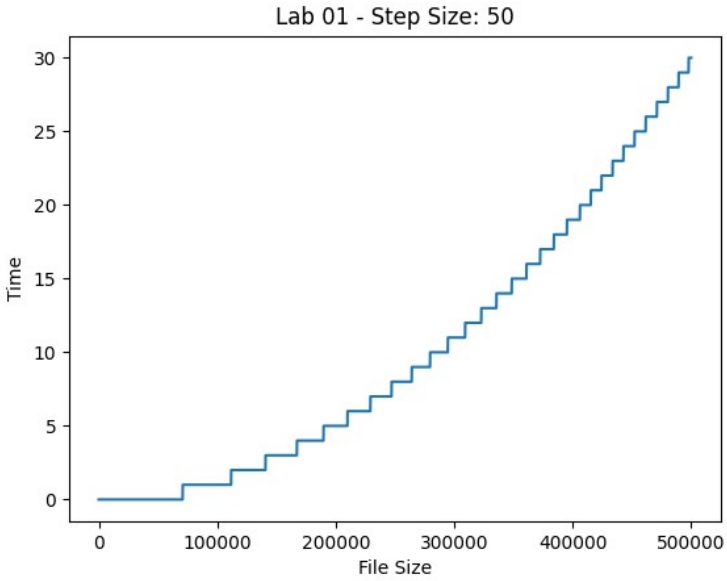
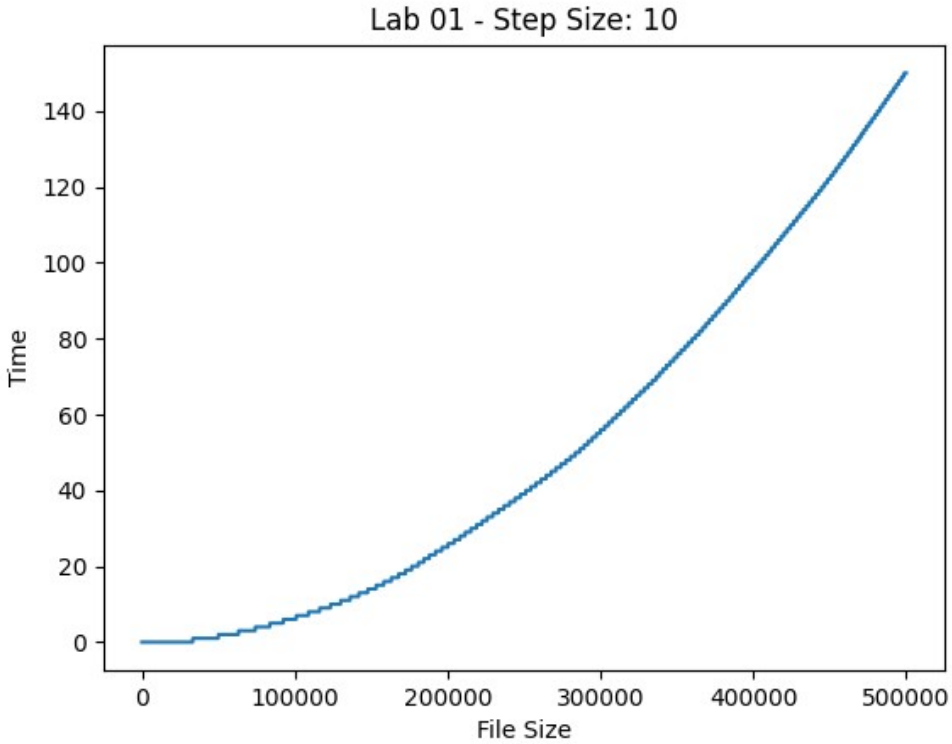
Diego Meseguer

COEN 146 – Computer Networks
Lab 01 Report

Graphs:







We can see from the three graphs that the time to copy the file is growing exponentially as the file size increases. In the graph with step size 100 we can see that the path of the line has a square shape as we are increasing the writing in chunks of 100. We can see that the performance starts at a good point with a horizontal line at the level of time 0 that extends further than the other graphs but then it starts to increase sharply and the intervals where the line is horizontal become shorter. With step size 50 we can see more clearly the shape of exponential growth. The times are worse than the first graph that has step size 100 since we've gone from max time 16 to max time 30 and the start portion before reaching file size 100,000 is not as good since we go above time 0 sooner. Finally, with step size 10 we can see how the performance clearly worsens and we reach times as high as 140. Thus, we can see from the three graphs that a bigger step size when copying files gives us better speed.

I think that the reason for the look of the graphs based on the step size is that when we use a smaller step size we are writing more data to the destination file compared to what we would write if we have used a larger step size. This is something that can be noted when looking at the file sizes of the resulting destination files for different step sizes and how they can be gigabytes bigger when one goes from step size 100 to the slower step size 10.

To implement the graphs we used the matplotlib Python module. In order to feed the data from the C program into the Python script we used Unix pipes. So the command that was used is

./file_copier 2 | python plot.py

In this way, file_copier uses printf() to output the data to standard output. Then plot.py reads this data from standard input using input() and generates the plots using matplotlib. file_copier had to be modified a little bit for this to work by hard-coding the values of the step size and max file size instead of asking the user.

Finally, for the implementation of file_copier both C function calls and system calls were used. Thus, there are two copier functions: copier_with_functions_calls and copier_with_sys_calls that do the same thing and can be used in place of the other.