Queuing Theory

Question 1:

The average length of the queue is 8.1 people.

The average wait time in the queue is 162.0 seconds.

Question 2:

The average length of the queue is 0.888888888889 people.

The average wait time in the queue is 1.33333333333 seconds.

The probability of having 5 users in the system is 0.0658436213992

Question 3:

The average length of the queue is 4.05 people.

The average wait time in the queue is 27.0 seconds.

The average wait time in the queue is 33.0 seconds.

Queuing Analysis

For this section of the project we used a python script that would download a file 100 times. We would take the time before and after the download and then store the difference. After downloading all 100 times we would take the average time and set mu as 1 over the average. We did this for both the lighttpd server and the web-server we built in lab 4. In order to actual plot the graphs used a lambda from 0 to 100 and plotted rho as lambda over mu as our x coordinate and 1 over mu minus lambda as our y coordinate

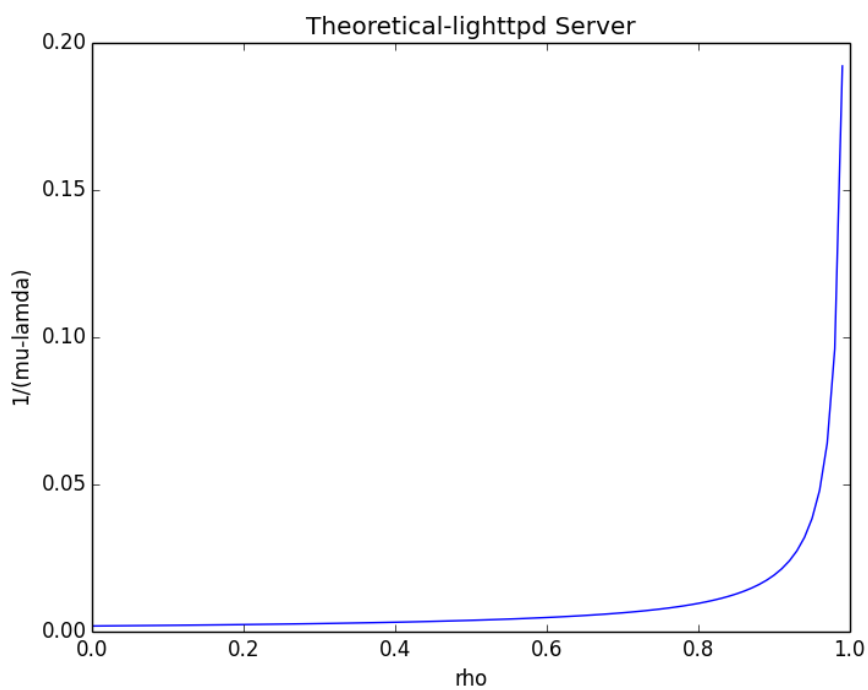

fig 1

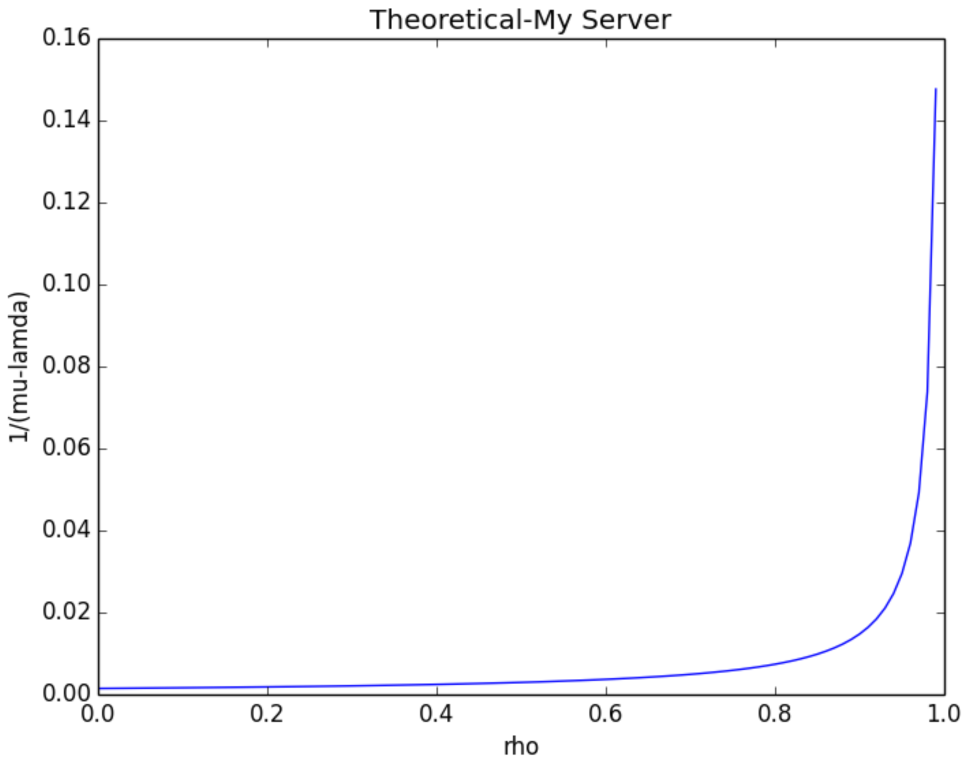

fig 2

As you can see in figures 1 and 2, using the average download time of a file on both servers provides a nice exponential curve just as you would expect.

Performance Evaluation

In this section of the lab we used another python script to run the generator script provided for us by Dr. Zappala. We broke the theoretical graph into ten chunks and then used the generator script to produce workloads at this level. We stored the data in text files and then plotted the results on the over the theoretical points . As you can see in figure 3, the lighttpd server initially averaged below the prediction but still had significant portions of the outliers above predicted time. As the load increased the lighttpd server actual seemed to decrease it's time and the tail of the outliers shrunk. This could have been due to optimization done by the server or already having the files in memory. In figure 4 you can see that the results of lab server does not appear as consistent. It's outlier tail is not as compact and has many outliers not even close to the rest of the data points throughout the entire graph whereas in figure 3 you really only have three points of this nature and they are all at the higher end of the workload spectrum. The lab server might be more susceptible to variations in the operating system or network. This would make sense considering that the lab server was created in a couple of days and the lighttpd server has been around for over a decade. Considering this fact, the difference between the two results is astounding because they are so similar. While it might be possible that in a much larger more real world test case, the lighttpd server would show significant superiority to the lab server, in this test case they are almost identical. We ran the test several times to insure that the results were in fact valid, and with only minor fluctuations be between each iteration of the tests we achieved the same results each time.

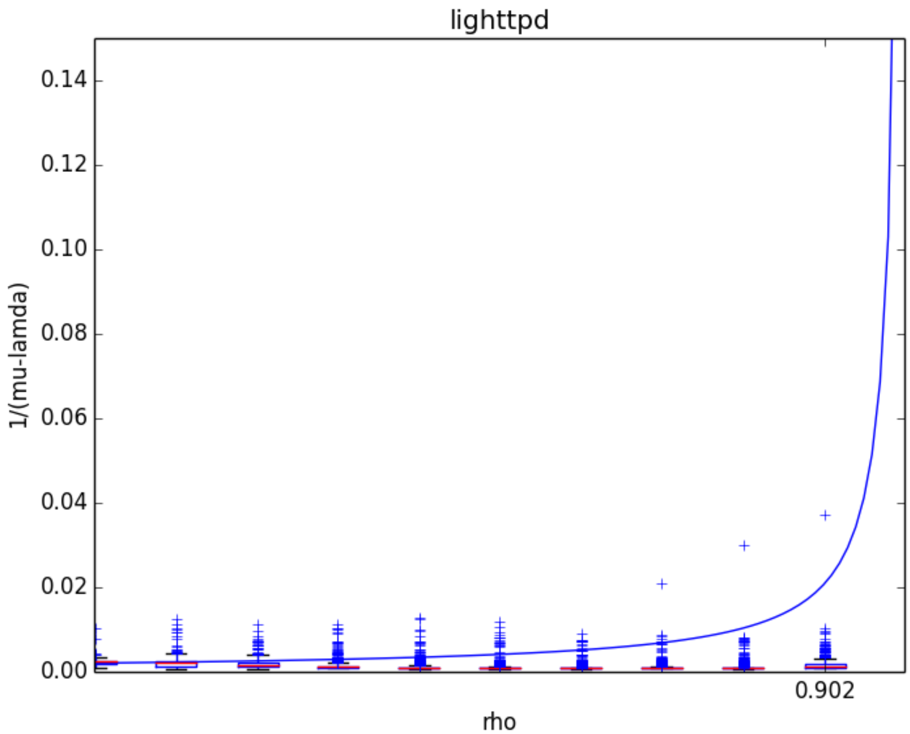

fig 3

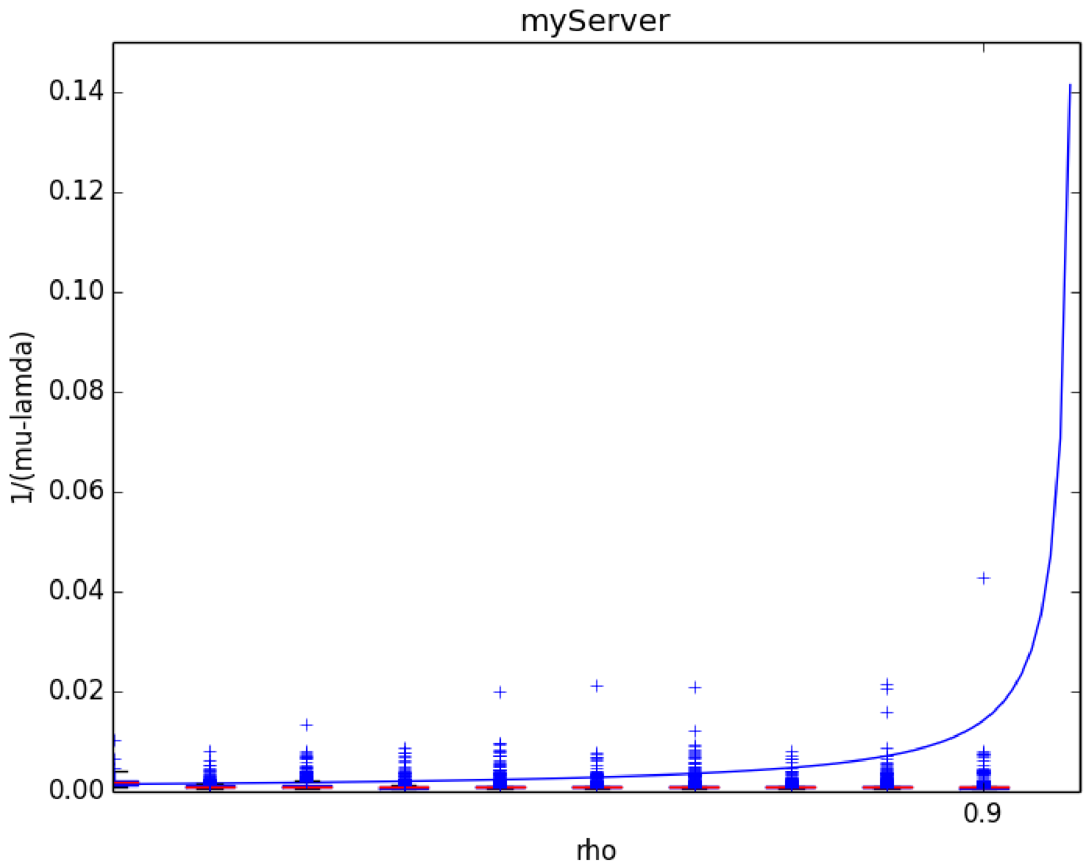

fig 4

Conclusion

After doing this lab we arrived at several conclusions. The first being that most of the speed and efficiency in the internet as more to do with the lower layers like IP and TCP then it does with the actual server or high level program requesting or sending the network traffic. While we knew is in theory from lectures, we were able to see the actual data and because we built the one of the servers, we knew that it was not professional grade(our so we thought).

To explore this further there are several things we would like to try, we would like to implement the other http commands and see if we would achieve similar results on all the other commands like post, put and so on. We would also like to look into why the lighttpd server appears to be more consistent. Why did our server have more erratic outliers and what would it take to reduce those.