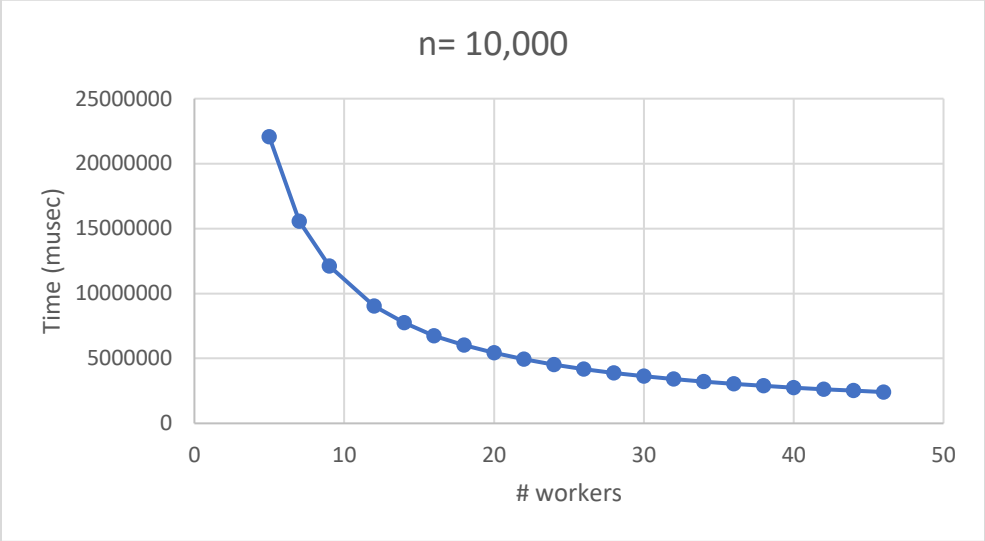
Sakshi Choudhary
CSCE 313 – PA 3
Report

1. In this assignment, I am increasing the efficiency of the client-server program through multithreading. I am creating w amount of worker threads that process requests made by the three request threads. This is different from the initial code because the original code only uses one worker thread to process the n amount of requests.
2. The given client code is much slower with n= 10,000 because it only has one worker thread versus the modified client code which can have many worker threads. For example 10,000 with one worker thread took around 110 seconds and 861955 microseconds whereas with 5 worker threads it took 22 seconds and 56208 microseconds long. As w increases the time for the program to finish decreases but then eventually stabilizes. However, the overhead eventually outweighs the benefits at this point because as the you approach the CPU, Memory, and I/O bounds the program goes through many more context switches and utilizes more mechanisms like locking and unlocking which slows performance.

w	**time (musec)**
5	22056208
7	15566116
9	12101281
12	9023862
14	7750825
16	6745857
18	6021401
20	5418467
22	4930968
24	4528495
26	4183326
28	3888280
30	3642353
32	3401961
34	3219722
36	3040380
38	2888540
40	2750028
42	2619964
44	2515404
46	2408326



3. The platform on which I gathered your timing data was the CSCE Linux server**.** The maximum number of threads allows was 46. When trying to use 47 worker threads there would be an error reported. The error : DATASERVER: control: pthread_create failure: Resource temporarily unavailable , data47_:CLIENT:cread: broken/closed pipe detected, exiting thread...: File exists. When trying to create more threads than allows, the OS throws an error and processes as many requests as it can. The client program responds by exiting the thread and not processing all the requests but reporting the output of whatever it processed.