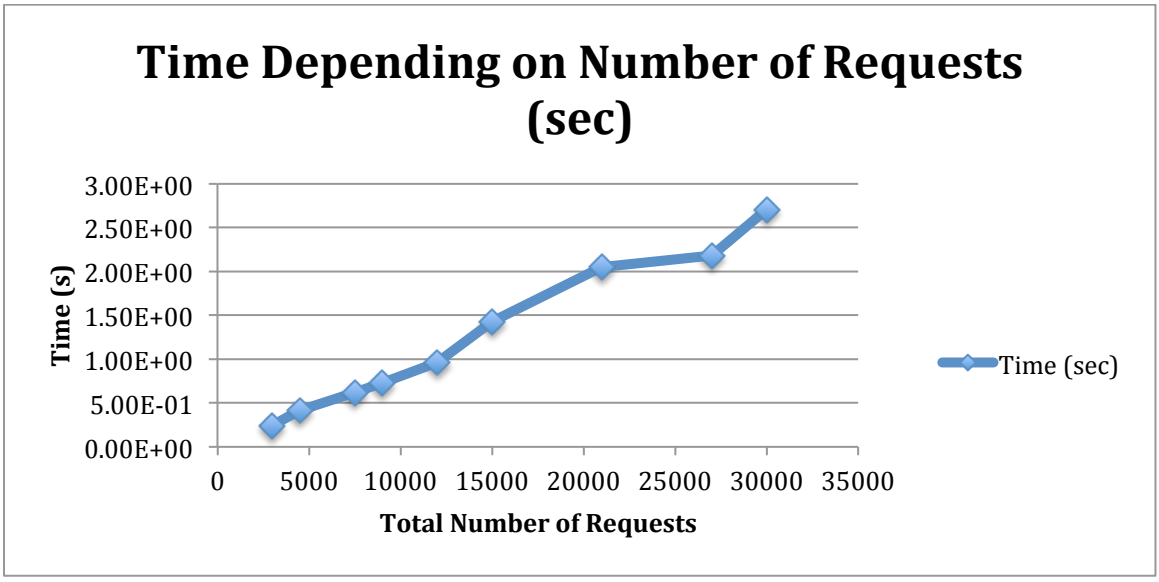
Jonathan Westerfield
Bettati
224005649

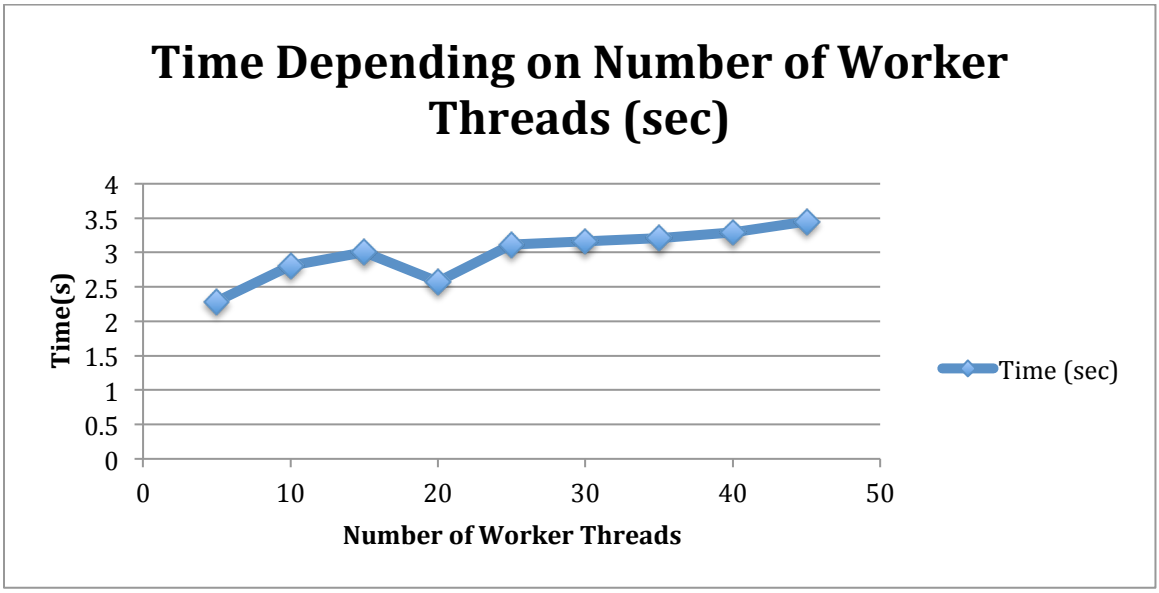
Machine Problem 3 Analysis

I ran all of my tests on my machine. When changing the number of requests, I got the results shown below. The test was performed with 15 worker threads and a bounded request buffer of size 500. As you can see, as the number of requests are increased, the time it takes to complete all of the requests (provided everything else stays the same), will increase in a linear fashion.



The next set of tests focused on the time changes whenever changing the number of worker threads present. The tests were performed with 30000 requests (1000 per request thread) and a bounded request buffer of size 500.When changing the number of worker threads. As you can see from the graph below, increasing the number of worker threads did not change the amount of time it took to complete the tasks in any significant way. This is because of the fact that there is only 1 bounded buffer shared between all of the threads. Because of this and the usage of semaphore for the bounded buffer, each thread was forced to access the buffer sequentially, instead of concurrently. This created a bottleneck in the program and will prevent any significant improvements to the time, no matter how many worker threads are used with it. In fact, looking at the trend of the graph, it could also be assumed that the more worker threads are created, the slower the program will run. This is because of the overhead of creating and running a thread coupled with the problem of a single request buffer.

Jonathan Westerfield
Bettati
224005649



The next set of tests dealt with changing the size of the bounded buffer. The tests were performed with 30000 requests (10000 per request thread) and 15 worker threads. As you can see from the graph below, increasing the size of the request buffer only made dramatic improvements whenever the size was small. The improvement continues for a little of the graph after a spike. However, the time spikes again and then returns to normal. This means the size of the bounded buffer is really inconsequential to the run time of the program. The spikes in time may mean that there are some other underlying factors being done by the operating system between each run. NOTE: my program would not join the worker threads when it went past 45 worker threads; it simply just hung.