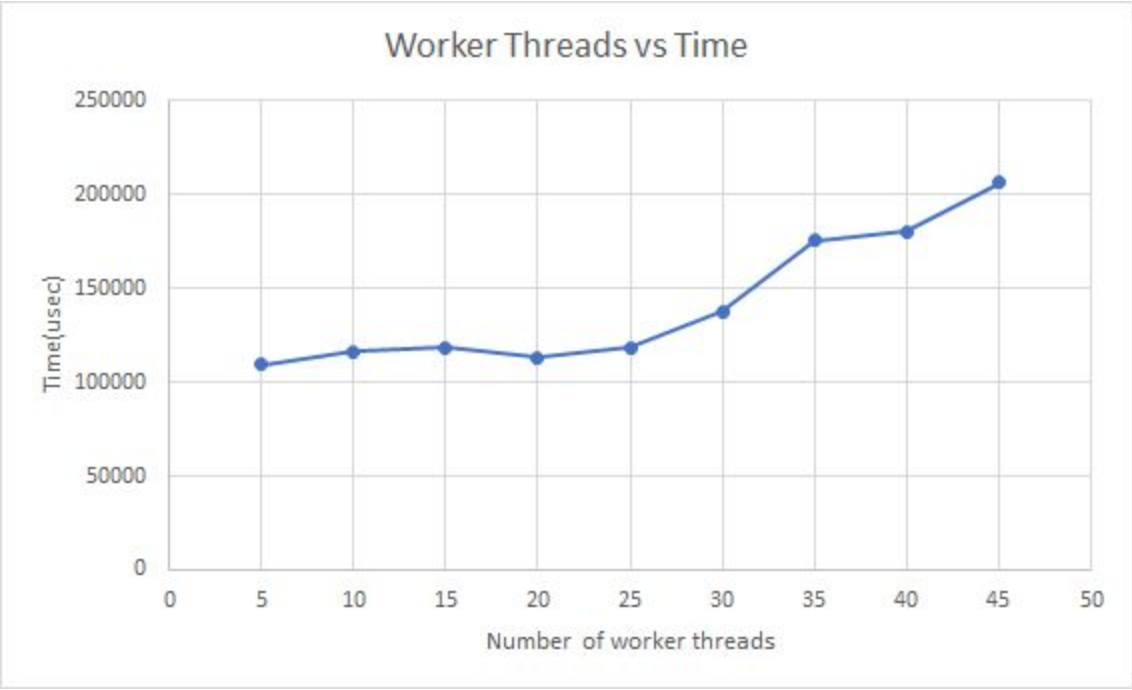
NUMBER OF THREADS

So according to the data below it seems as though increasing the number of worker threads does not improve the efficiency of the code. In fact it looks like the more you increase the amount of worker threads, the longer is takes for the to run through the program. I think this is due to the fact that there is only one pc buffer used for all of the threads, in my code it is called work_buff. Since the code only used one pc buffer, the threads cannot run in “parallel” they have to run one after another. So this ended up created a bottleneck that prevent the program from improving in efficiency by increasing the amount of worker threads.

-n 10 -b 5 -w??

Number of worker threads	Time (usec)
5	109542
10	116209
15	118236
20	113199
25	118310
30	137830
35	175476
40	180052
45	206334



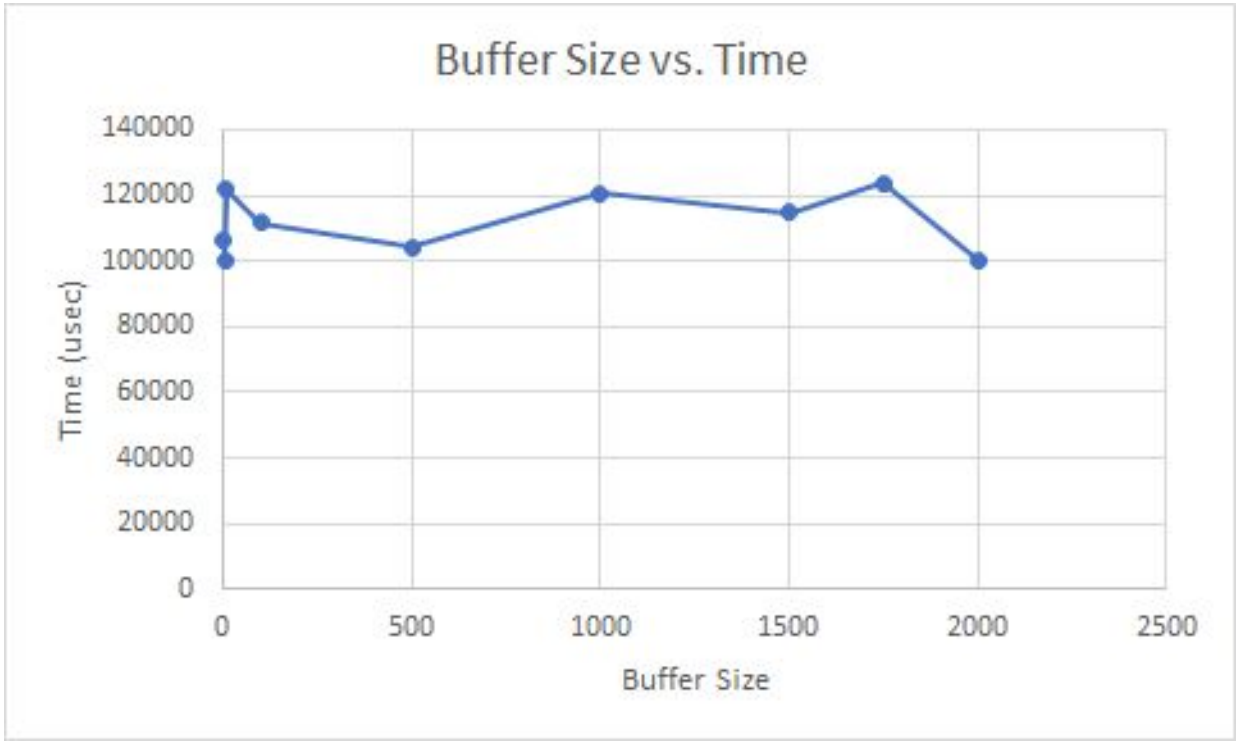
BUFFER SIZE

According to the data below it seems as though adjuisting the buffer size does not greatly change the runtime of the program. There are little dips and and spikes but I think those occur simply due to some other unknown factors.

-n 10 -b ?? - w 20

Buffer size	Time(usec)
1	106463
5	100263
10	121998

100	111823
500	104356
1000	120716
1500	114975
1750	123816
2000	100434



Note:

Example of code running on my pc