Program Structures and Algorithms
Spring 2024

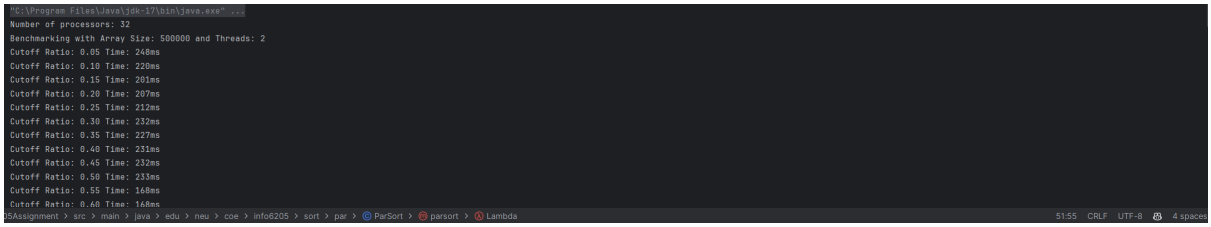
NAME: Gautham Venkata Krishna Prasad
NUID: 002249901
GITHUB LINK: https://github.com/gauthamkris7neu/INFO6205Assignment

Task:

Assignment 5 (Parallel Sorting)

Relationship Conclusion:

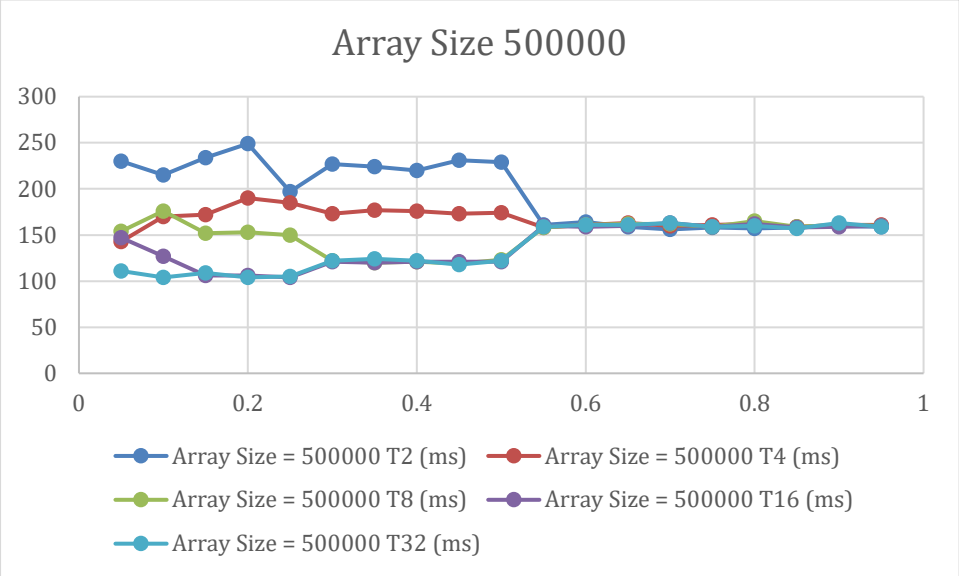
From the data we got from the benchmark runs, it appears that the parallel sorting algorithm demonstrates improved performance with increasing thread count up to a certain point. The data indicates that using this sorting algorithm yields a decrease in sorting time with an increase in threads, conforming to a trend where $T = K/t$ for $t \leq p$ (where T is time, t is the number of threads , K is a constant, and p is the number of processors). However, this benefit does not scale linearly, and once the number of threads exceeds the number of processors ($t > p$), the time reduction flattens out due to overhead, suggesting an optimal thread count close to the physical limit of the processor.

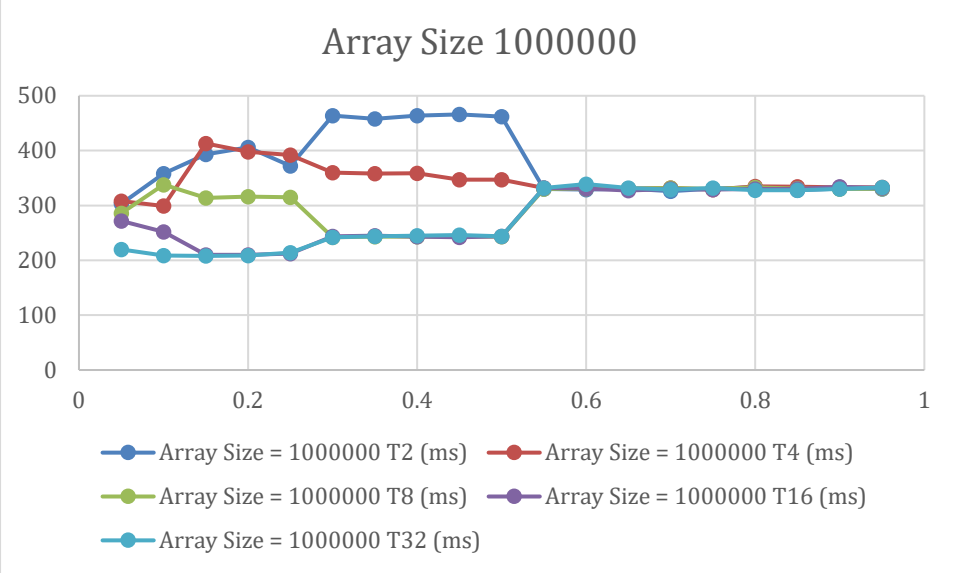


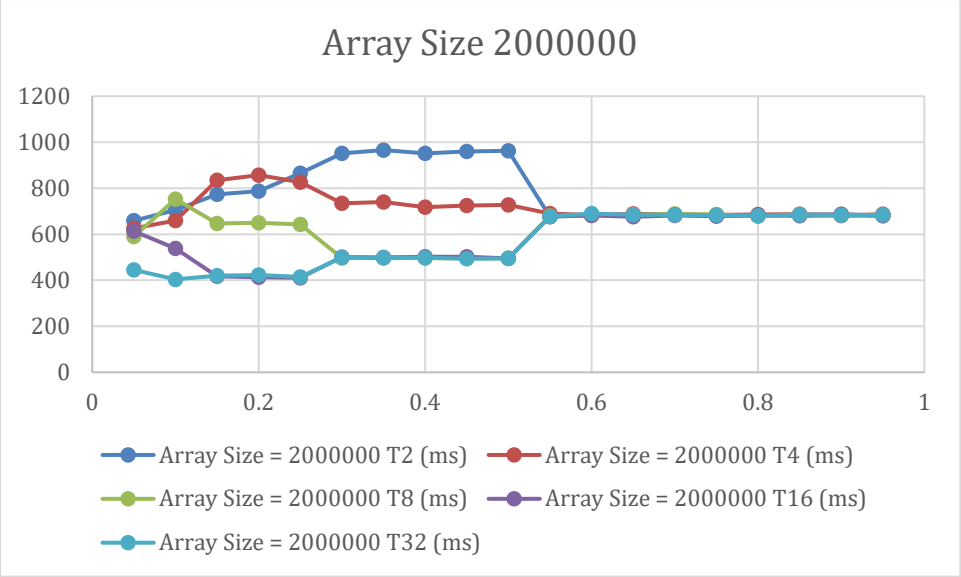
Evidence to support that conclusion:

	Array Size = 500000					Array Size = 1000000					Array Size = 2000000				
Cutoff Ratio	T2 (ms)	T4 (ms)	T8 (ms)	T16 (ms)	T32 (ms)	T2 (ms)	T4 (ms)	T8 (ms)	T16 (ms)	T32 (ms)	T2 (ms)	T4 (ms)	T8 (ms)	T16 (ms)	T32 (ms)
0.05	230	143	154	147	111	303	308	286	272	220	659	626	590	613	445
0.1	215	170	176	127	104	358	299	338	252	209	706	660	753	538	404
0.15	234	172	152	106	109	393	413	314	210	208	773	835	647	417	420
0.2	249	190	153	106	104	406	398	316	210	209	788	857	650	413	423
0.25	197	185	150	104	105	372	392	315	212	214	866	826	643	411	414
0.3	227	173	122	121	122	464	360	242	244	242	952	734	497	501	499
0.35	224	177	120	120	124	458	358	243	245	244	966	740	497	499	498
0.4	220	176	121	121	122	464	359	243	243	245	952	718	497	502	498
0.45	231	173	119	121	118	466	347	244	242	246	960	725	495	502	494
0.5	229	174	123	121	122	462	347	243	244	244	963	728	495	495	495
0.55	161	158	158	160	159	330	332	330	332	332	676	690	680	677	677
0.6	164	161	161	159	161	329	332	331	331	339	685	683	683	682	690
0.65	159	163	162	160	161	331	331	331	327	332	682	688	685	676	684
0.7	156	160	162	163	163	326	332	332	330	329	685	686	688	682	685
0.75	158	161	159	159	159	331	329	330	329	332	680	685	686	679	683

0.8	157	162	165	162	160	333	335	333	330	328	680	686	679	683	680
0.85	158	159	159	158	157	328	334	331	330	328	680	687	684	685	686
0.9	160	161	161	159	163	331	333	330	334	330	681	682	686	687	683
0.95	159	161	160	160	159	330	333	331	332	333	680	687	682	681	685






Array Size 2000000
1200
1000
800
600
400
200
0
0
0.2
0.4
0.6
0.8
1
Array Size = 2000000 T2 (ms)
Array Size = 2000000 T4 (ms)
Array Size = 2000000 T8 (ms)
Array Size = 2000000 T16 (ms)
Array Size = 2000000 T32 (ms)