Vipin Mamidi (NUID 001582139)

# Program Structures & Algorithms

# FALL 2021

# Assignment No. 5 (Parallel Sorting)

## ▪ Task

Your task is to implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. You will consider two different schemes for deciding whether to sort in parallel.

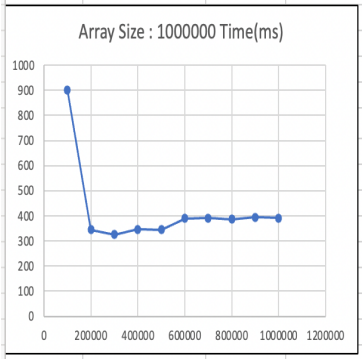
1. A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running. It's your job to experiment and come up with a good value for this cutoff. If there are fewer elements to sort than the cutoff, then you should use the system sort instead.
2. Recursion depth or the number of available threads. Using this determination, you might decide on an ideal number (*t*) of separate threads (stick to powers of 2) and arrange for that number of partitions to be parallelized (by preventing recursion after the depth of *lg t* is reached).
3. An appropriate combination of these.

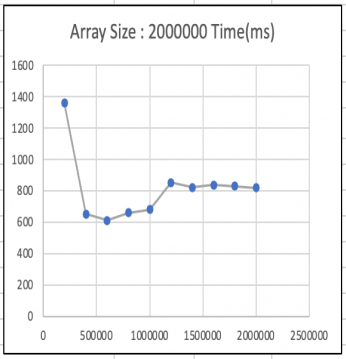
## ▪ Result With Evidence

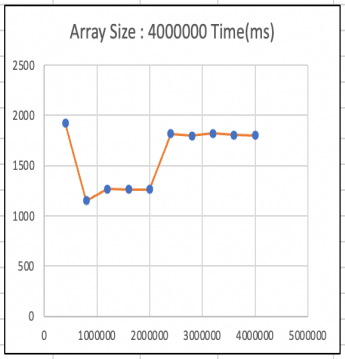
**1.**

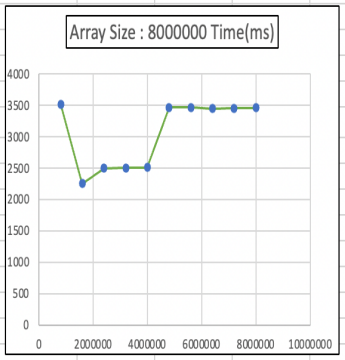
**Degree of Parallelism : 32**

| Array Size : 1000000 | | Array Size : 2000000 | | Array Size : 4000000 | | Array Size : 8000000 | |
|---|---|---|---|---|---|---|---|
| Cut Off | Time(ms) | Cut Off | Time(ms) | Cut Off | Time(ms) | Cut Off | Time(ms) |
| 100000 | 901 | 200000 | 1359 | 400000 | 1921 | 800000 | 3518 |
| 200000 | 344 | 400000 | 651 | 800000 | 1150 | 1600000 | 2253 |
| 300000 | 326 | 600000 | 611 | 1200000 | 1266 | 2400000 | 2498 |
| 400000 | 346 | 800000 | 659 | 1600000 | 1265 | 3200000 | 2504 |
| 500000 | 344 | 1000000 | 679 | 2000000 | 1263 | 4000000 | 2508 |
| 600000 | 389 | 1200000 | 853 | 2400000 | 1818 | 4800000 | 3465 |
| 700000 | 391 | 1400000 | 821 | 2800000 | 1798 | 5600000 | 3467 |
| 800000 | 387 | 1600000 | 837 | 3200000 | 1823 | 6400000 | 3445 |
| 900000 | 394 | 1800000 | 830 | 3600000 | 1804 | 7200000 | 3450 |
| 1000000 | 391 | 2000000 | 819 | 4000000 | 1800 | 8000000 | 3458 |



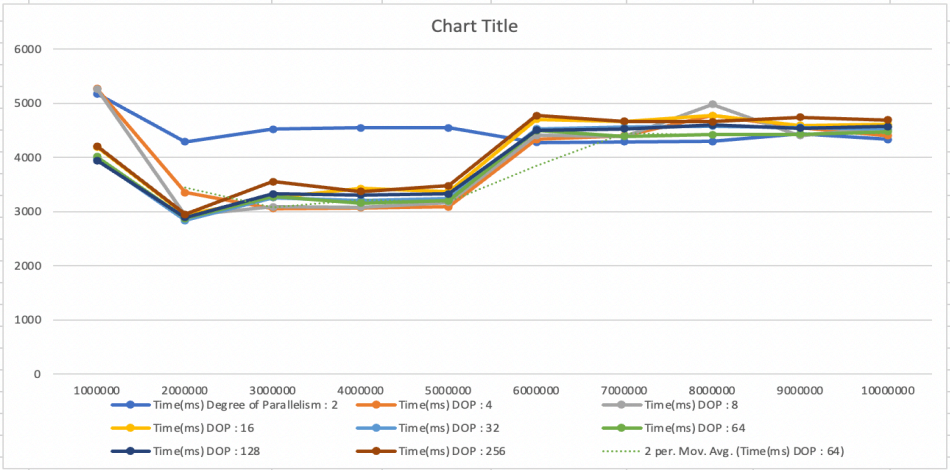




From the above data and chart where with different array sizes ranging from 1000000 to 8000000, and the count of threads (degree of parallelism) is constant that is 32 for all the experiments, it can be observed that there is a decrease of time at first, reach the lowest point and then increase a bit where cutoff value is around 20% of the array size in all the experiments. Thus, concluding that 20% of the array size is the best cutoff value.

## 2.

| Array Size : 10000000 | Time(ms) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Cut Off | Degree of Parallelism : 2 | DOP : 4 | DOP : 8 | DOP : 16 | DOP : 32 | DOP : 64 | DOP : 128 | DOP : 256 |
| 1000000 | 5177 | 5267 | 5266 | 4194 | 3977 | 4011 | 3944 | 4206 |
| 2000000 | 4288 | 3354 | 2924 | 2936 | 2840 | 2874 | 2896 | 2952 |
| 3000000 | 4523 | 3055 | 3089 | 3261 | 3258 | 3279 | 3328 | 3555 |
| 4000000 | 4544 | 3069 | 3080 | 3428 | 3208 | 3156 | 3307 | 3371 |
| 5000000 | 4544 | 3086 | 3169 | 3374 | 3241 | 3200 | 3325 | 3479 |
| 6000000 | 4277 | 4330 | 4410 | 4706 | 4526 | 4496 | 4506 | 4775 |
| 7000000 | 4286 | 4401 | 4388 | 4662 | 4562 | 4392 | 4526 | 4664 |
| 8000000 | 4293 | 4773 | 4981 | 4775 | 4581 | 4422 | 4597 | 4663 |
| 9000000 | 4431 | 4548 | 4405 | 4594 | 4554 | 4426 | 4542 | 4744 |
| 10000000 | 4332 | 4404 | 4563 | 4610 | 4523 | 4479 | 4570 | 4693 |




From the above data and chart where the array size is 10000000, and the count of threads (degree of parallelism) is increased with power 2 from 2 to 256, It can be observed that all the threads varied little until 8 threads performed, after that in all threads performance the time is not varied much even until degree of parallelism is 256.

By Comparing and considering all the experiments, it can be concluded that the optimum cutoff is 20% of array size and the optimum number of threads is 8 (as per mac book air). And it is worth to choose parallel sorting as it saves the time.