**INFO 6205**

**Program Structures & Algorithms**

**Fall 2020**

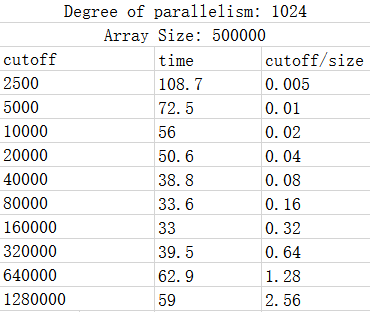
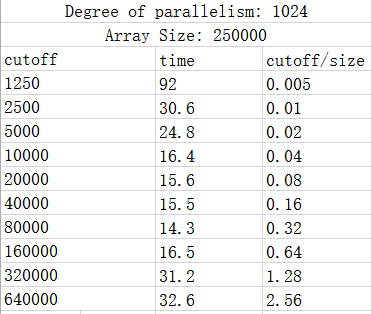
**Assignment No 5**

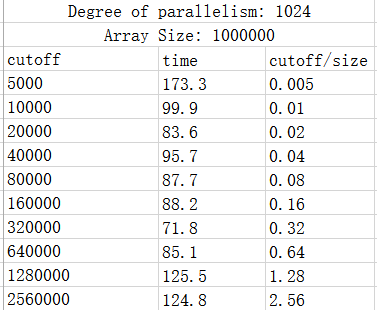
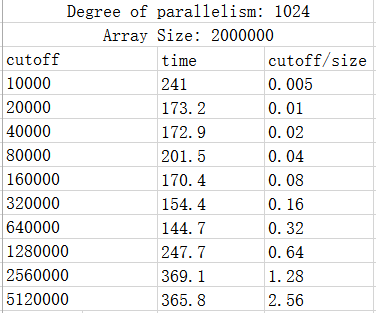
* **Task**

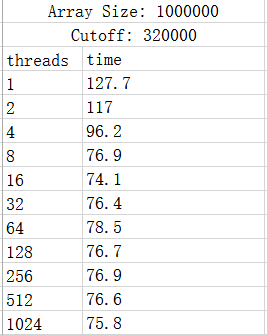
To implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. Consider two different schemes for deciding whether to sort in parallel.

1. A cutoff (defaults to, say, 1000) which will be updated according to the first argument in the command line when running. To experiment and come up with a good value for this cutoff. If there are fewer elements to sort than the cutoff, then use the system sort instead.
2. Recursion depth or the number of available threads. Using this determination, 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).

* **Output** (few outputs to prove relationship)

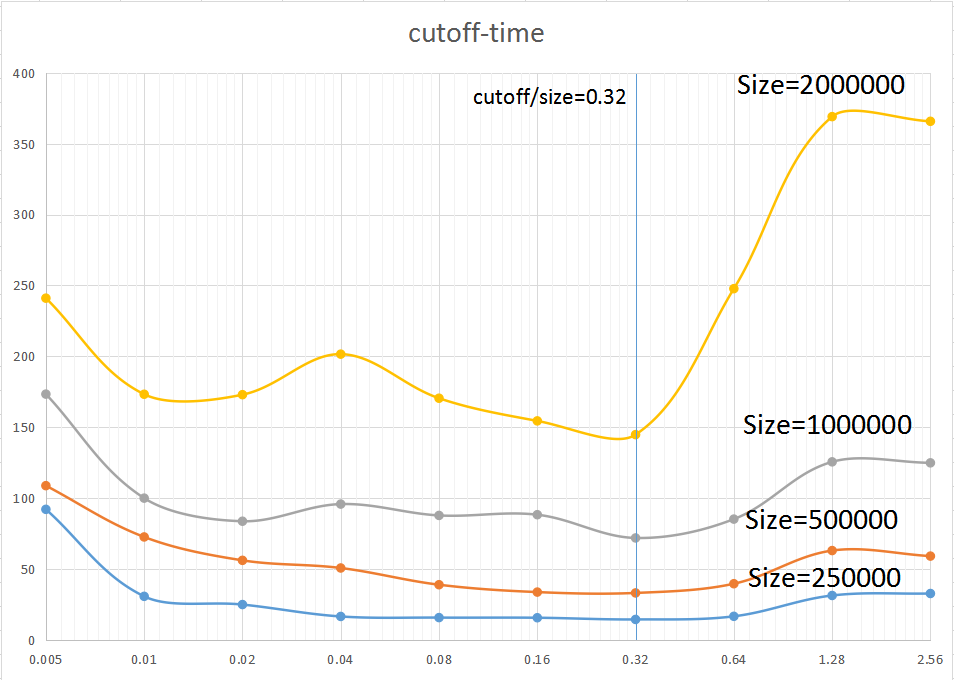
For the scheme of cutoff and array size, I set the degree of parallelism as 1024, then change size and cutoff multiple times to test the running time. The results are shown below:



For the scheme of threads, an ideal number of separate threads should stick to powers of 2 since it is decided by the recursion depth. I test the running time when the array size is 1000000 and cutoff is 320000, which is 32% of array size.

* **Relationship conclusion and **Evidence****

1. The “cutoff-time chart” obtained from the results is shown below:

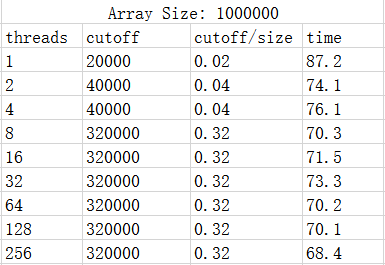


From the chart, I found that for each size of array, as the cutoff increases, the time decreases at the beginning and then increases after a value. Therefore, it can be concluded that the lowest time point is about where cutoff is about 32% of array size.

1. The “threads-time chart” obtained from results is shown below:

From the chart, I found that running time decreases before 10 while it doesn’t change a lot after 10. Thus, the conclusion is when thread increases from 1-8, running time will decrease, after 8, time will not change.

Besides, the optimum cutoff varies according to the number of threads, the minimal time of different number of threads is shown below:



The optimum cutoff shown in form increases at first and then becomes stable when it comes to 8 threads.

Thus, the optimum cutoff is 32% of array size and the optimum number of threads is 8.