**INFO 6205**

**Program Structures & Algorithms**

**Fall 2018**

**Assignment 4-Report**

Parallel Sorting is a sorting approach in which we sort the inputs in a sequence simultaneously to increase the efficiency and performance of the algorithm.

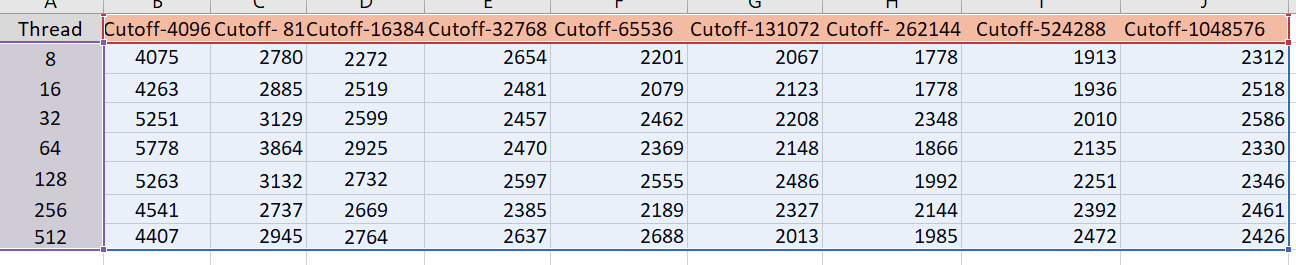
1. **Observation:**

* The program uses multithreading to achieve the result.
* Degree of parallelism depends on the processor of the machine.
* The CompletableFuture.supplyAsync is a method which by default invokes ForkJoinPool.commonPool() to create thread, while we recursively call sort function.
* We can also create and limit the number of thread by ExecutorService.
* While the size of array is not exceeding the cutoff non-parallel merge sort is called else we are diving into parallel sort.

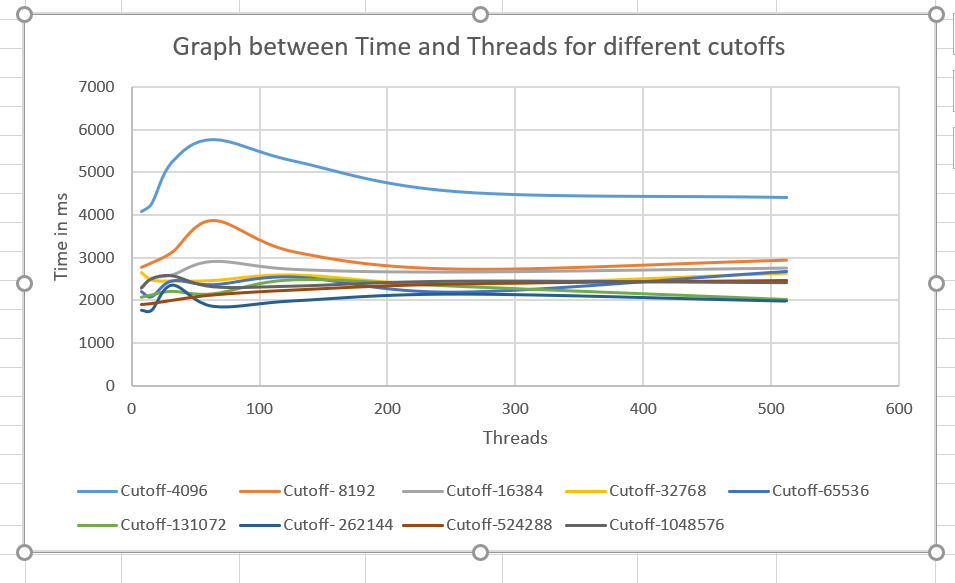
**1. Conclusion:**

* Degree of parallelism in my machine is 7.
* As the cutoff increases the time with it increases, but by using multithreading it lowers the time because we are doing simultaneous sorting.
* The value of thread is taken as the power of 2 for more precise outputs.
* The cutoff value which is least is T(n) = 1778 for cutoff value 262144.

1. **Graphical Representation** -



The Graphs for the above values are as follows:



1. **Proof:**

This method is better than the arrays.sort method as we are performing multithreading by which task gets divided and provide better time results. The time of the array sort depends on two factors cutoff and number of thread.

From the above experiment conducted we can say the best cutoff for the above data is 262144, for 16 or 32 threads, as the result time is least for it.