Program Structures and Algorithms

Spring 2023(SEC –03)

NAME: Yufan Gao

NUID: 002755691

**Task:**

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.

**Relationship Conclusion:**

When the ratio for cutoff/array.length is between 0.06-0.50, it consumes time the least.

The efficacy keeps decreasing until the threads become 8. After threads count becomes 8, it arrives the most efficient point and starts to be stable.

**Evidence to support that conclusion:**

I found that the consuming time is related to cutoff/array.length. Thus, I made two experiments as the graphs show in the latter part.

For threads, I chose the ratio of cutoff/array.length to be 0.3 since it conforms to least time consuming. Then I start to test the relationship between efficacy and threads’ number.

**Graphical Representation:**











