Haoyang Hu (001005743)

**Program Structures & Algorithms**

**Fall 2021**

**Assignment No. 05**

* **Task**

Please see the presentation on *Assignment on Parallel Sorting* under the *Exams. etc.*module.

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.

* **Relationship Conclusion:**

1. From the output below, when the number of threads is bigger than 8 or less than 8, the process time is longer obviously, so there is no necessary to assume one more appropriate data, the most efficient threads number is 8.

2. According to the graph and output, on my computer, I think the best combination of thread and cutoff is that uses 8 threads, and the value of cutoff is less than 10% of the size of the array, and nearly 50% of the size of the array is the worst.

* **Evidence to support the conclusion:**

1. **Output**

**experiment:**

**图片包含 应用程序

描述已自动生成**

**电脑屏幕的照片上有文字

描述已自动生成**

**文本

描述已自动生成**

**8 threads are the best:**

**文本

描述已自动生成**

**文本

描述已自动生成**

**文本

描述已自动生成**

**文本

中度可信度描述已自动生成**

1. **Graphical Representation**

图表, 折线图

描述已自动生成