Yuan Lian (NUID 001063563)

**Program Structures & Algorithms**

**FALL 2021**

**Assignment No. 5**

* **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.

**Part 1 Output:**

**一些文字和图案

中度可信度描述已自动生成**

**Graphical Representation:**

图表

描述已自动生成

**Result:**

Though beginning with much time, the parSort became really stable when cutoff increase. Between range 130000 to 175000, parSort is faster. The fastest came in 8 threads when cutoff is 149000 parSort sort 2000000 numbers in 694ms.

**图表

低可信度描述已自动生成 表格

中度可信度描述已自动生成**

**Part 2 Output:**

**日历

中度可信度描述已自动生成**

**Graphical Representation:**

**图形用户界面, 直方图

描述已自动生成**

**Result:**

Though some unstable outcomes happened, in general, as threads number increase for computing, ParSort became faster. But there are two stages, before 130000 and after 130000. Passing 1300000 after, the thread took more time on average than before 130000.

**Relationship Conclusion:**

More thread we used, the higher speed ParSort have. When the ratio of cutoff and array sizes are between 0.04-0.13, the Parsort will have highest efficiency.

**Evidence to support the conclusion:**

More thread. First, with fixed array size for partition, more threads we use, faster the speed is. As a result, I changed array size for different partition. When array size increase, the time also increase. And the relation is linear. ParSort will more efficient when the ratio of cutoff and array sizes are between 0.04-0.13. It also showed in result in part 1.

**Output:**

表格

描述已自动生成

**Graphical Representation:**

图形用户界面, 应用程序

描述已自动生成