002133899 Kaifeng Ruan

**Program Structures & Algorithms**

**Fall 2021**

**Assignment No. 5**

* **Task (List down the tasks performed in the Assignment)**

Array size from 2^21 ~ 2^24

Threads number from 2 to 8

Cut off from 2^13 to 2^20

* **Relationship Conclusion:**

In order to measure the desired cut off and number of threads, I used different array sizes, cut offs and number of threads. And they were all the power of 2 since the parSort is to divide the array into 2 part and sort recursively.

After I had collected all the data, I found that when I use 8 threads, the sort method uses least time. Since my computer has at most 9 threads for usage, I made an assumption that the more threads you use in the method, the faster the function will run.

And I use different values of cut offs from 2^13 to 2^20. The reason I use these numbers is that these numbers which are power of 2 can divide the array exactly into 2 pieces, and it will make the sorting efficiently.

By comparing different set of data I collected, I found that cut off between 2^16 to 2^19 can make the sorting cost less time. And arrays of smaller size runs faster with a cut off of 2^15, while arrays of bigger size runs faster with a cut off of 2^19. I think the proper number of cut off is related to the size of the array and the ratio between cut off and array size might be 1/32.

* **Evidence to support the conclusion:**

1. **A picture containing text

   Description automatically generatedOutput (Snapshot of Code output in the terminal)**

**A screenshot of a computer

Description automatically generated**

1. **Graphical Representation(Observations from experiments should be tabulated and analyzed by plotting graphs(usually in excel) to arrive on the relationship conclusion)**

**Array size = 2^21, Threads = 2**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **1632** |
| **2^14** | **742** |
| **2^15** | **531** |
| **2^16** | **542** |
| **2^17** | **588** |
| **2^18** | **682** |
| **2^19** | **823** |
| **2^20** | **857** |

**Array size = 2^21, Threads = 4**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **788** |
| **2^14** | **529** |
| **2^15** | **520** |
| **2^16** | **522** |
| **2^17** | **539** |
| **2^18** | **601** |
| **2^19** | **645** |
| **2^20** | **591** |

**Array size = 2^21, Threads = 8**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **755** |
| **2^14** | **545** |
| **2^15** | **528** |
| **2^16** | **515** |
| **2^17** | **538** |
| **2^18** | **533** |
| **2^19** | **500** |
| **2^20** | **601** |

**Array size = 2^22, Threads = 2**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **1474** |
| **2^14** | **1179** |
| **2^15** | **1156** |
| **2^16** | **1093** |
| **2^17** | **1124** |
| **2^18** | **1213** |
| **2^19** | **1426** |
| **2^20** | **1624** |

**Array size = 2^22, Threads = 4**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **1522** |
| **2^14** | **1149** |
| **2^15** | **1104** |
| **2^16** | **1084** |
| **2^17** | **1128** |
| **2^18** | **1155** |
| **2^19** | **1222** |
| **2^20** | **1311** |

**Array size = 2^22, Threads = 8**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **1637** |
| **2^14** | **1185** |
| **2^15** | **1175** |
| **2^16** | **1071** |
| **2^17** | **1047** |
| **2^18** | **1043** |
| **2^19** | **1062** |
| **2^20** | **998** |

**Array size = 2^23, Threads = 2**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **4877** |
| **2^14** | **3334** |
| **2^15** | **2830** |
| **2^16** | **2537** |
| **2^17** | **2471** |
| **2^18** | **2391** |
| **2^19** | **2561** |
| **2^20** | **3022** |

**Array size = 2^23, Threads = 4**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **1522** |
| **2^14** | **1149** |
| **2^15** | **1104** |
| **2^16** | **1084** |
| **2^17** | **1128** |
| **2^18** | **1155** |
| **2^19** | **1222** |
| **2^20** | **1311** |

**Array size = 2^23, Threads = 8**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **5119** |
| **2^14** | **3536** |
| **2^15** | **3001** |
| **2^16** | **2700** |
| **2^17** | **2473** |
| **2^18** | **2294** |
| **2^19** | **2227** |
| **2^20** | **2281** |

**Array size = 2^24, Threads = 2**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **12515** |
| **2^14** | **6032** |
| **2^15** | **5762** |
| **2^16** | **5641** |
| **2^17** | **4675** |
| **2^18** | **4435** |
| **2^19** | **4513** |
| **2^20** | **4893** |

**Array size = 2^24, Threads = 4**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **8629** |
| **2^14** | **6786** |
| **2^15** | **5729** |
| **2^16** | **5174** |
| **2^17** | **4537** |
| **2^18** | **4491** |
| **2^19** | **4289** |
| **2^20** | **4885** |

**Array size = 2^24, Threads = 8**

|  |  |
| --- | --- |
| **Cut Off** | **Time(ms)** |
| **2^13** | **9043** |
| **2^14** | **7365** |
| **2^15** | **5723** |
| **2^16** | **5505** |
| **2^17** | **4727** |
| **2^18** | **4218** |
| **2^19** | **4083** |
| **2^20** | **4477** |