Program Structures and Algorithms
Spring 2024

NAME: Ashish Nevan Gade
NUID: 002889005
GITHUB LINK: https://github.com/AshishNevan/INFO6205

Task: Assignment 1: Parallel Sort result and efficacy

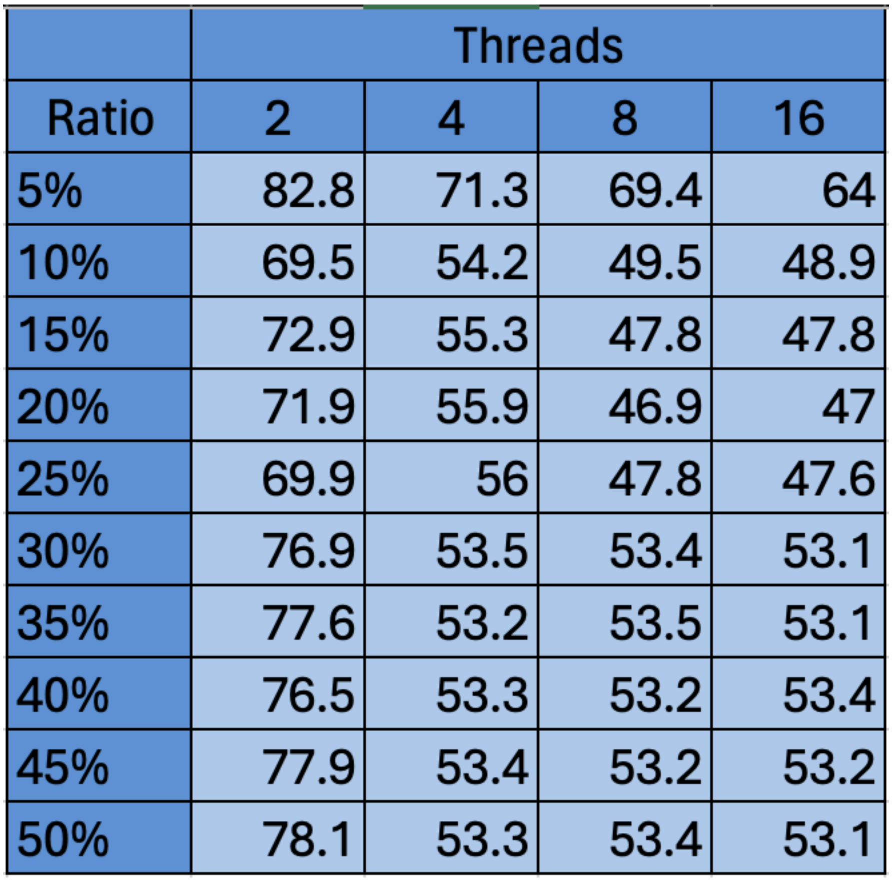
Relationship Conclusion:

For recursion depth (d) and number of threads (t):

$$t = 2^{d}$$

Evidence to support that conclusion:

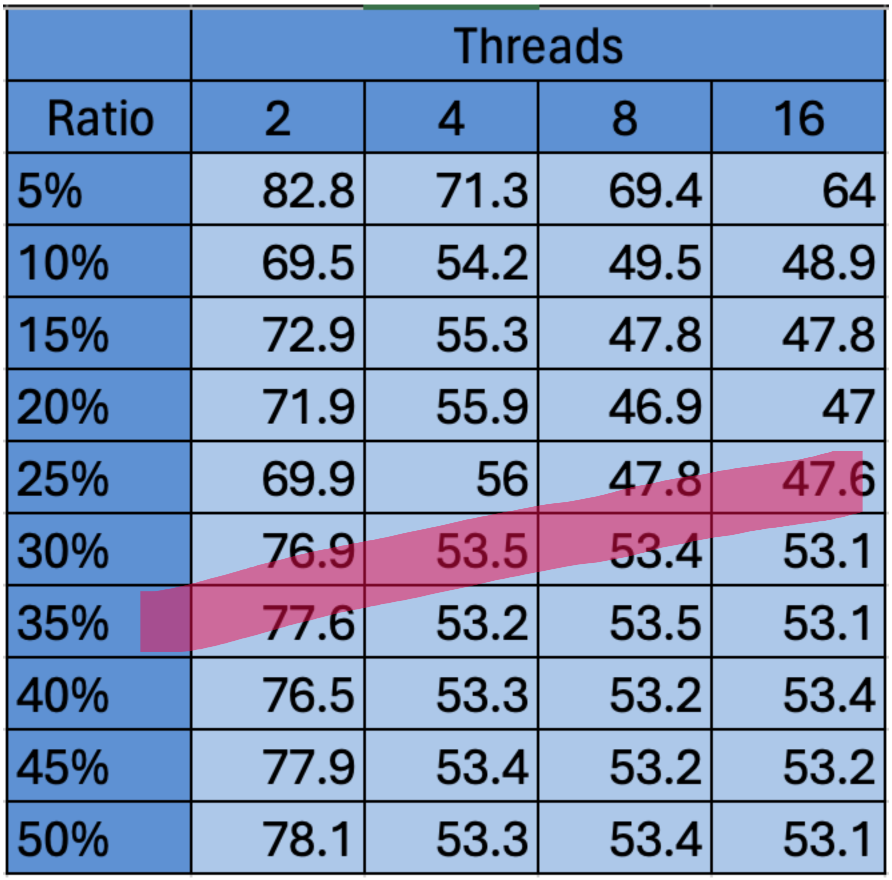
The runtimes of 10 iterations of parallel sort on array size (N) = 2Million, with different combinations of threads and cutoff ratios are listed:



	Threads			
Ratio	2	4	8	16
5%	82.8	71.3	69.4	64
10%	69.5	54.2	49.5	48.9
15%	72.9	55.3	47.8	47.8
20%	71.9	55.9	46.9	47
25%	69.9	56	47.8	47.6
30%	76.9	53.5	53.4	53.1
35%	77.6	53.2	53.5	53.1
40%	76.5	53.3	53.2	53.4
45%	77.9	53.4	53.2	53.2
50%	78.1	53.3	53.4	53.1

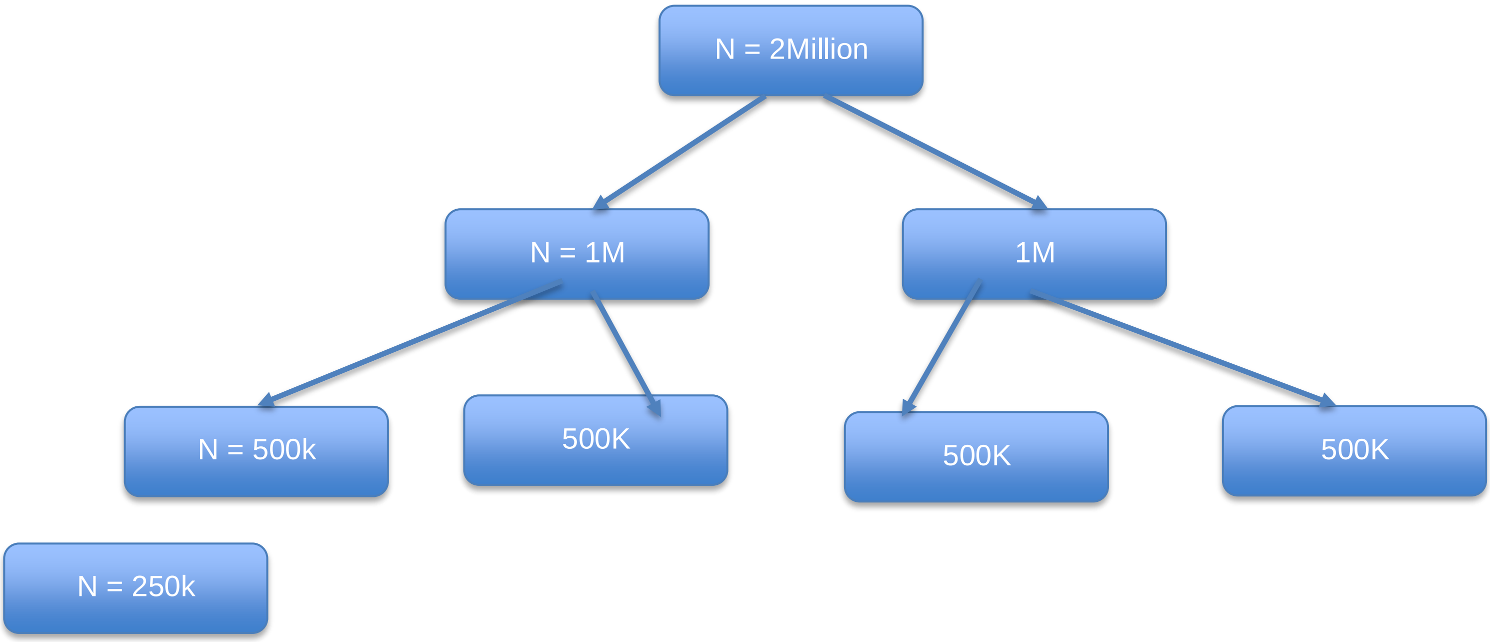
We can infer that lower cutoff values lead to higher recursion depth(parallel) and higher cutoff values lead to lower recursion depth. In order to handle many parallel sort, we will be needing more threads.

Through this experiment, we aim to find the "sweet-spot" of that combination of thread count and cutoff ratio which has the most performance. From the values shown in the table above, it is safe to say that out inference holds true. The "sweet-spot" of threads vs cutoff ratio is found along this diagonal shown below.



	Threads			
Ratio	2	4	8	16
5%	82.8	71.3	69.4	64
10%	69.5	54.2	49.5	48.9
15%	72.9	55.3	47.8	47.8
20%	71.9	55.9	46.9	47
25%	69.9	56	47.8	47.6
30%	76.9	53.5	53.4	53.1
35%	77.6	53.2	53.5	53.1
40%	76.5	53.3	53.2	53.4
45%	77.9	53.4	53.2	53.2
50%	78.1	53.3	53.4	53.1

The system I used to run this experiment has 8 threads, and for peak performance, we ought to use all 8 of those threads to perform computation parallelly. Let us attempt to do exactly that visually.

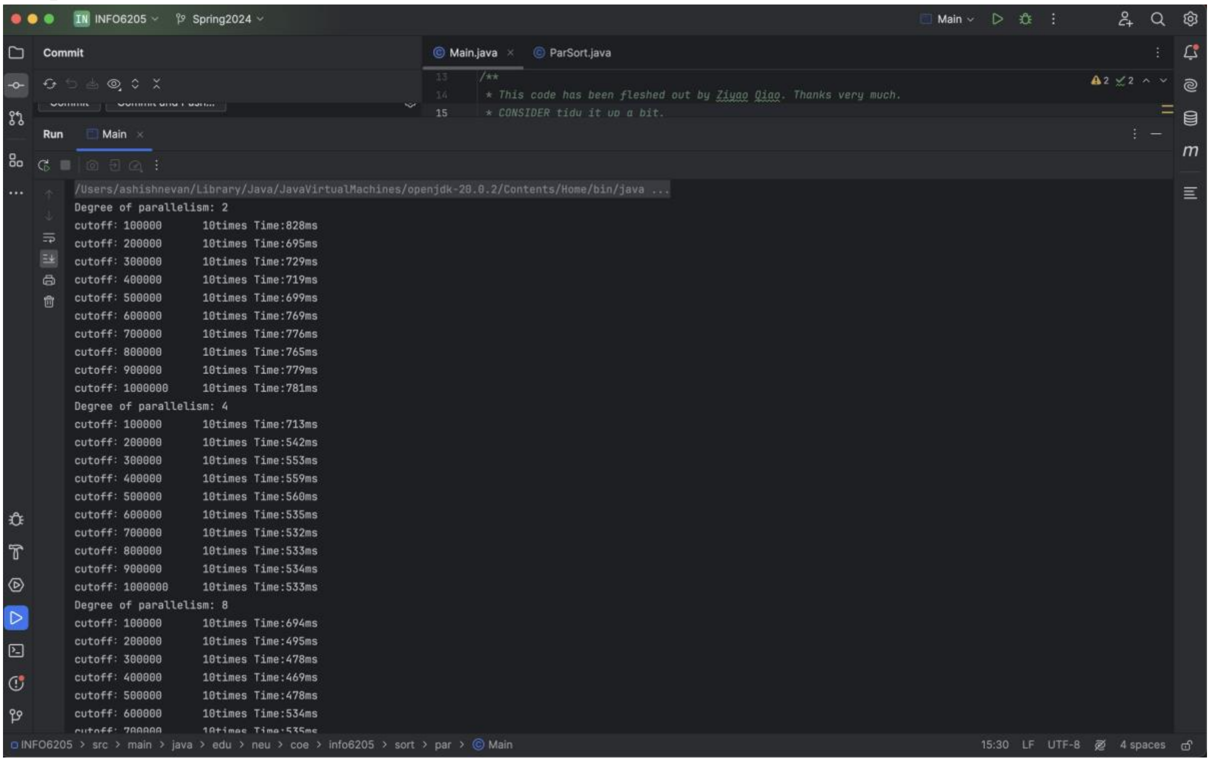


Cutoff of 400K (20% of 2M) has created 8 substasks (recursive depth 3) which would ideally fill 8 threads, an even lower cutoff would make the tasks overload the number of threads hence lead to tasks waiting for ready threads.

We can conclude from this experiment that parallelization parameters of parallel sort is dependent on number of physical threads available in the system. The relationship of max depth (d), arry size (N) and cutoff (c) is given by:

$$d = \lg(N/C)$$

Output:



INFO6205
Spring2024
Main
Commit
Main.java
ParSort.java
Run
Main
INFO6205 › src › main › java › edu › neu › coe › info6205 › sort › par › Main
15:30
LF
UTF-8
4 spaces

```
/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * CONSIDER tidy it up a bit.
```

```
/Users/ashishnevan/Library/Java/JavaVirtualMachines/openjdk-20.0.2/Contents/Home/bin/java ...
Degree of parallelism: 2
cutoff: 100000      10times Time:828ms
cutoff: 200000      10times Time:695ms
cutoff: 300000      10times Time:729ms
cutoff: 400000      10times Time:719ms
cutoff: 500000      10times Time:699ms
cutoff: 600000      10times Time:769ms
cutoff: 700000      10times Time:776ms
cutoff: 800000      10times Time:765ms
cutoff: 900000      10times Time:779ms
cutoff: 1000000     10times Time:781ms
Degree of parallelism: 4
cutoff: 100000      10times Time:713ms
cutoff: 200000      10times Time:542ms
cutoff: 300000      10times Time:553ms
cutoff: 400000      10times Time:559ms
cutoff: 500000      10times Time:560ms
cutoff: 600000      10times Time:535ms
cutoff: 700000      10times Time:532ms
cutoff: 800000      10times Time:533ms
cutoff: 900000      10times Time:534ms
cutoff: 1000000     10times Time:533ms
Degree of parallelism: 8
cutoff: 100000      10times Time:694ms
cutoff: 200000      10times Time:495ms
cutoff: 300000      10times Time:478ms
cutoff: 400000      10times Time:469ms
cutoff: 500000      10times Time:478ms
cutoff: 600000      10times Time:534ms
cutoff: 700000      10times Time:535ms
```