Zeynep Tufekci (NUID: 001565591)

# INFO 6205

## Program Structures & Algorithms

## Fall 2020

## Assignment 5

### Task and Observations

N : 10000, 100000, 200000, 500000, 1000000, 2000000
Thread Count : 4, 16, 64, 374, 1024
Cut off : $2^8$(256), $2^{10}$(1024), $2^{12}$(4096) $2^{14}$(16384), $2^{16}$(1048576)

### Output

```
Available Processor: 4
                N = 100000
                Thread Count: 4
Cutoff : 1024                   Average Time:24ms
Cutoff : 4096                   Average Time:17ms
Cutoff : 16364                  Average Time:15ms
Cutoff : 65536                  Average Time:14ms
                Thread Count: 16
Cutoff : 1024                   Average Time:11ms
Cutoff : 4096                   Average Time:8ms
Cutoff : 16364                  Average Time:6ms
Cutoff : 65536                  Average Time:5ms
                Thread Count: 64
Cutoff : 1024                   Average Time:8ms
Cutoff : 4096                   Average Time:8ms
Cutoff : 16364                  Average Time:5ms
Cutoff : 65536                  Average Time:6ms
                Thread Count: 256
Cutoff : 1024                   Average Time:12ms
Cutoff : 4096                   Average Time:6ms
Cutoff : 16364                  Average Time:6ms
Cutoff : 65536                  Average Time:5ms
                Thread Count: 1024
Cutoff : 1024                   Average Time:19ms
Cutoff : 4096                   Average Time:8ms
Cutoff : 16364                  Average Time:5ms
Cutoff : 65536                  Average Time:6ms

For N= 100000,  16364 and 65536 cut-off values give the best performance. Thread
counts higher than 4 give the best result. Cut off Rates(Cutoff/Number) are 0.16
and 0.64.

                N = 200000
                Thread Count: 4
Cutoff : 1024                   Average Time:21ms
Cutoff : 4096                   Average Time:13ms
Cutoff : 16364                  Average Time:12ms
Cutoff : 65536                  Average Time:11ms
Cutoff : 161144                 Average Time:21ms
                Thread Count: 16
Cutoff : 1024                   Average Time:21ms
Cutoff : 4096                   Average Time:16ms
Cutoff : 16364                  Average Time:15ms
Cutoff : 65536                  Average Time:13ms
Cutoff : 161144                 Average Time:15ms
```

Zeynep Tufekci (NUID: 001565591)

```
                Thread Count: 64
Cutoff : 1024                   Average Time:21ms
Cutoff : 4096                   Average Time:16ms
Cutoff : 16364                  Average Time:14ms
Cutoff : 65536                  Average Time:14ms
Cutoff : 161144                 Average Time:26ms
                Thread Count: 256
Cutoff : 1024                   Average Time:37ms
Cutoff : 4096                   Average Time:14ms
Cutoff : 16364                  Average Time:13ms
Cutoff : 65536                  Average Time:18ms
Cutoff : 161144                 Average Time:15ms
                Thread Count: 1024
Cutoff : 1024                   Average Time:43ms
Cutoff : 4096                   Average Time:16ms
Cutoff : 16364                  Average Time:14ms
Cutoff : 65536                  Average Time:12ms
Cutoff : 161144                 Average Time:15ms

For N= 200000 Cut-off value 65536 is the best value among them.

                N = 500000
                Thread Count: 4
Cutoff : 1024                   Average Time:55ms
Cutoff : 4096                   Average Time:47ms
Cutoff : 16364                  Average Time:48ms
Cutoff : 65536                  Average Time:46ms
Cutoff : 161144                 Average Time:35ms
                Thread Count: 16
Cutoff : 1024                   Average Time:63ms
Cutoff : 4096                   Average Time:39ms
Cutoff : 16364                  Average Time:38ms
Cutoff : 65536                  Average Time:37ms
Cutoff : 161144                 Average Time:37ms
                Thread Count: 64
Cutoff : 1024                   Average Time:62ms
Cutoff : 4096                   Average Time:40ms
Cutoff : 16364                  Average Time:37ms
Cutoff : 65536                  Average Time:40ms
Cutoff : 161144                 Average Time:39ms
                Thread Count: 256
Cutoff : 1024                   Average Time:62ms
Cutoff : 4096                   Average Time:44ms
Cutoff : 16364                  Average Time:87ms
Cutoff : 65536                  Average Time:96ms
Cutoff : 161144                 Average Time:75ms
                Thread Count: 1024
Cutoff : 1024                   Average Time:133ms
Cutoff : 4096                   Average Time:93ms
Cutoff : 16364                  Average Time:328ms
Cutoff : 65536                  Average Time:306ms
Cutoff : 161144                 Average Time:119ms

For 500000 number of elements, less thread number (4 and 16) give the best result
with 35 ms. When thread number is increasing, performance is getting better in
small cut-off values. For example, cutoff value 4096 is better than other
options.
```

Zeynep Tufekci (NUID: 001565591)

```
                N = 1000000
                Thread Count: 4
Cutoff : 1024                   Average Time:256ms
Cutoff : 4096                   Average Time:185ms
Cutoff : 16364                  Average Time:163ms
Cutoff : 65536                  Average Time:182ms
Cutoff : 161144                 Average Time:144ms


                Thread Count: 16
Cutoff : 1024                   Average Time:266ms
Cutoff : 4096                   Average Time:184ms
Cutoff : 16364                  Average Time:183ms
Cutoff : 65536                  Average Time:173ms
Cutoff : 161144                 Average Time:178ms
                Thread Count: 64
Cutoff : 1024                   Average Time:252ms
Cutoff : 4096                   Average Time:184ms
Cutoff : 16364                  Average Time:172ms
Cutoff : 65536                  Average Time:194ms
Cutoff : 161144                 Average Time:185ms
                Thread Count: 256
Cutoff : 1024                   Average Time:169ms
Cutoff : 4096                   Average Time:107ms
Cutoff : 16364                  Average Time:105ms
Cutoff : 65536                  Average Time:113ms
Cutoff : 161144                 Average Time:99ms
                Thread Count: 1024
Cutoff : 1024                   Average Time:160ms
Cutoff : 4096                   Average Time:115ms
Cutoff : 16364                  Average Time:98ms
Cutoff : 65536                  Average Time:90ms
Cutoff : 161144                 Average Time:111ms

For N is 1000000, cut off value 161144 is the best value among them. Performance
is better in higher thread counts. For example 256 and 1024 thread counts has the
best time performance.

                N = 2000000
                Thread Count: 4
Cutoff : 1024                   Average Time:352ms
Cutoff : 4096                   Average Time:231ms
Cutoff : 16364                  Average Time:175ms
Cutoff : 65536                  Average Time:253ms
Cutoff : 161144                 Average Time:384ms
Cutoff : 1048576                Average Time:522ms
                Thread Count: 16
Cutoff : 1024                   Average Time:611ms
Cutoff : 4096                   Average Time:315ms
Cutoff : 16364                  Average Time:311ms
Cutoff : 65536                  Average Time:335ms
Cutoff : 161144                 Average Time:335ms
Cutoff : 1048576                Average Time:329ms
                Thread Count: 64
Cutoff : 1024                   Average Time:422ms
Cutoff : 4096                   Average Time:360ms
Cutoff : 16364                  Average Time:359ms
Cutoff : 65536                  Average Time:353ms
Cutoff : 161144                 Average Time:352ms
Cutoff : 1048576                Average Time:382ms
                Thread Count: 256
Cutoff : 1024                   Average Time:475ms
Cutoff : 4096                   Average Time:359ms
Cutoff : 16364                  Average Time:344ms
```

Zeynep Tufekci (NUID: 001565591)

```
Cutoff : 65536                  Average Time:311ms
Cutoff : 161144                 Average Time:321ms
Cutoff : 1048576                Average Time:360ms
                Thread Count: 1024
Cutoff : 1024                   Average Time:486ms
Cutoff : 4096                   Average Time:365ms
Cutoff : 16364                  Average Time:366ms
Cutoff : 65536                  Average Time:335ms
Cutoff : 161144                 Average Time:326ms
Cutoff : 1048576                Average Time:359ms
For N= 2000000, small cut off number is not good choice. Cut off value 65536 and
161144 have good performance. Also, Thread count 4 with cut off value 16364
outperform in this experiment. Cut off rate is 0.0008, 0.03 and 0.08.


This is also other experiment which I tried 1,2 and 4 thread count. Last 3 values
give the best performance whereas One thread count also performs closely better.

                N = 4000000
                Thread Count: 1
Cutoff : 1024                   Average Time:623ms
Cutoff : 4096                   Average Time:446ms
Cutoff : 16364                  Average Time:468ms
Cutoff : 65536                  Average Time:741ms
Cutoff : 161144                 Average Time:716ms
Cutoff : 1048576                Average Time:935ms
Cutoff : 1500000                Average Time:912ms
                Thread Count: 2
Cutoff : 1024                   Average Time:829ms
Cutoff : 4096                   Average Time:660ms
Cutoff : 16364                  Average Time:636ms
Cutoff : 65536                  Average Time:636ms
Cutoff : 161144                 Average Time:676ms
Cutoff : 1048576                Average Time:749ms
Cutoff : 1500000                Average Time:763ms
                Thread Count: 4
Cutoff : 1024                   Average Time:883ms
Cutoff : 4096                   Average Time:669ms
Cutoff : 16364                  Average Time:653ms
Cutoff : 65536                  Average Time:327ms
Cutoff : 161144                 Average Time:300ms
Cutoff : 1048576                Average Time:269ms
Cutoff : 1500000                Average Time:258ms
```

## Relationship conclusion

Parallel programming has good improvements in this experiment. Thread count between 4 to 1024 has close efficacy.
Cut off value less than 1024 degrade performance. Because of that, cut off values should be chosen more than 1024 for large numbers. Most of case, 16000 and 64000 cut off values outperform comparing to other values.

## Evidence to support relationship

Amdahl's Law is showed that it is possible to speedup program by running independent task with parallel computing.

$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

where

$S_{\text{latency}}$ is the theoretical speedup of the execution of the whole task,
$s$ is the number of threads across which the parallel portion is split,

Zeynep Tufekci (NUID: 001565591)

*p* is the proportion of execution time that the part benefiting from improved resources originally occupied.

Although it is possible get program faster by parallelizing, the speedup is limited by the serial part of the program. For example, if 95% of the program can be parallelized, the theoretical maximum speedup using parallel computing would be 20 times.

## *REFERENCES*

*Amdahl, Gene M. (1967). "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities" (PDF). AFIPS Conference Proceedings (30): 483–485. doi:10.1145/1465482.1465560.*

*Code part:*

```
int[] numberN = {100000, 200000, 500000, 1000000,2000000, 4000000 };
int[] threadCounts = { 1, 2, 4}; //, 16, 64, 256, 1024 };
int[] cutOffs = { 1024, 4096, 16364, 65536, 161144, 1048576, 1500000 };
Random random = new Random();
for (int n = 0; n < numberN.length; n++) {
    int[] array = new int[numberN[n]];
    System.out.println("        N = " + numberN[n]);
    ArrayList<Long> timeList = new ArrayList<>();

    for (int th = 0; th < threadCounts.length; th++) {
        ParSort.myPool = new ForkJoinPool(threadCounts[th]);
        System.out.println("        Thread Count: " + threadCounts[th]);

        for (int j = 0; j < cutOffs.length; j++) {
            if (cutOffs[j] > numberN[n])
                continue;
            ParSort.cutoff = cutOffs[j];

            long time;
            long startTime = System.currentTimeMillis();
            for (int t = 0; t < 10; t++) {
                for (int i = 0; i < array.length; i++)  array[i] = random.nextInt(10000000);
                ParSort.sort(array, 0, array.length);
            }
            long endTime = System.currentTimeMillis();
            time = (endTime - startTime);
            timeList.add(time);

            System.out.println("Cutoff:" + (ParSort.cutoff) + "\t\t Average Time:" + time / 10 + "ms");
        }
    }
}
```