

# Akshaysingh Bayes (NU ID: 002956209)

## 6205 - Program Structures and Algorithms

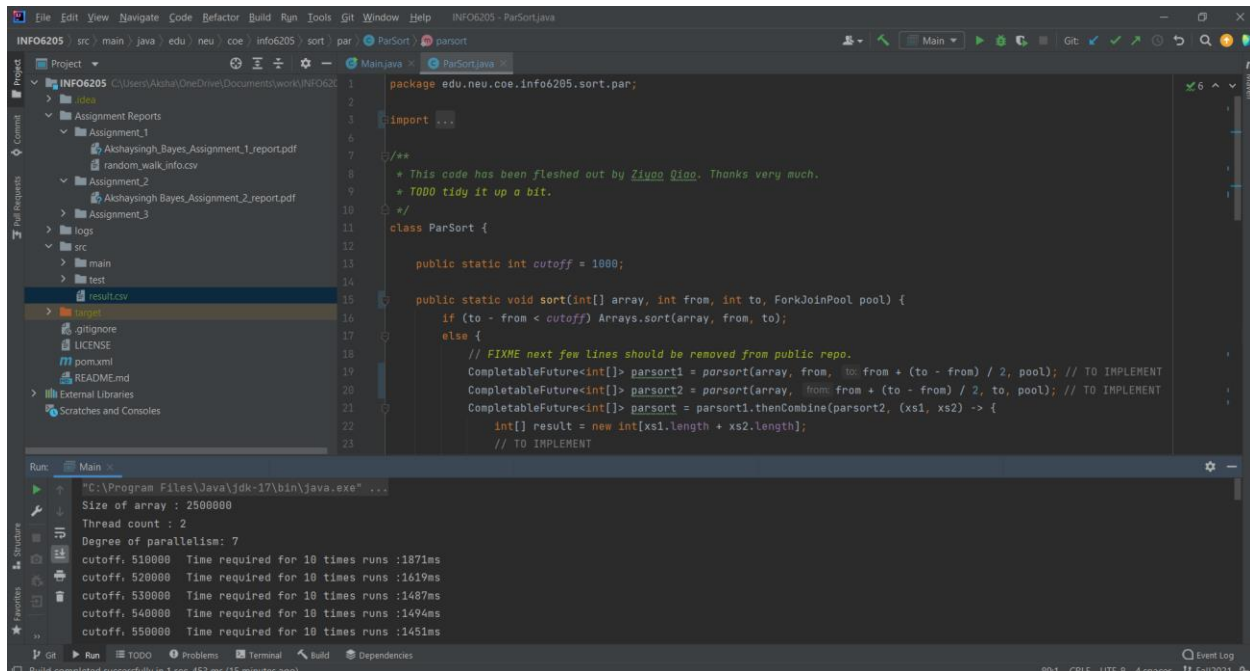
### Assignment - 5

#### Problem Statement:

To implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. Considering two different schemes for deciding whether to sort in parallel

- A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running.
- 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).

#### Output:



The screenshot shows an IDE with the following code in `ParSort.java`:

```
package edu.neu.coe.info6205.sort.par;

import ...

/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * TODO tidy it up a bit.
 */
class ParSort {

    public static int cutoff = 1000;

    public static void sort(int[] array, int from, int to, ForkJoinPool pool) {
        if (to - from < cutoff) Arrays.sort(array, from, to);
        else {
            // FIXME next few lines should be removed from public repo.
            CompletableFuture<int[]> parsort1 = parsort(array, from, to - from + (to - from) / 2, pool); // TO IMPLEMENT
            CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to, pool); // TO IMPLEMENT
            CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) -> {
                int[] result = new int[xs1.length + xs2.length];
                // TO IMPLEMENT
            });
        }
    }
}
```

The console output shows the following results:

```
Run: Main
"C:\Program Files\Java\jdk-17\bin\java.exe" ...
Size of array : 2500000
Thread count : 2
Degree of parallelism: 7
cutoff, 510000 Time required for 10 times runs :1871ms
cutoff, 520000 Time required for 10 times runs :1619ms
cutoff, 530000 Time required for 10 times runs :1487ms
cutoff, 540000 Time required for 10 times runs :1494ms
cutoff, 550000 Time required for 10 times runs :1451ms
```

#### Console Output:

Size of array : 2500000

Thread count : 2

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1871ms

cutoff : 520000 Time required for 10 times runs :1619ms

cutoff : 530000 Time required for 10 times runs :1487ms

cutoff : 540000 Time required for 10 times runs :1494ms

cutoff : 550000 Time required for 10 times runs :1451ms

cutoff : 560000 Time required for 10 times runs :1628ms

cutoff : 570000 Time required for 10 times runs :1709ms

cutoff : 580000 Time required for 10 times runs :1514ms

cutoff : 590000 Time required for 10 times runs :1524ms

cutoff : 600000 Time required for 10 times runs :1414ms

Thread count : 4

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1601ms

cutoff : 520000 Time required for 10 times runs :1487ms

cutoff : 530000 Time required for 10 times runs :1529ms

cutoff : 540000 Time required for 10 times runs :1540ms

cutoff : 550000 Time required for 10 times runs :1550ms

cutoff : 560000 Time required for 10 times runs :1544ms

cutoff : 570000 Time required for 10 times runs :1576ms

cutoff : 580000 Time required for 10 times runs :1554ms

cutoff : 590000 Time required for 10 times runs :1558ms

cutoff : 600000 Time required for 10 times runs :1542ms

Thread count : 8

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1236ms

cutoff : 520000 Time required for 10 times runs :1304ms

cutoff : 530000 Time required for 10 times runs :1253ms

cutoff : 540000 Time required for 10 times runs :1263ms

cutoff : 550000 Time required for 10 times runs :1267ms

cutoff : 560000 Time required for 10 times runs :1227ms

cutoff : 570000 Time required for 10 times runs :1278ms

cutoff : 580000 Time required for 10 times runs :1242ms

cutoff : 590000 Time required for 10 times runs :1286ms

cutoff : 600000 Time required for 10 times runs :1295ms

Thread count : 16

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :845ms

cutoff : 520000 Time required for 10 times runs :851ms

cutoff : 530000 Time required for 10 times runs :894ms

cutoff : 540000 Time required for 10 times runs :848ms

cutoff : 550000 Time required for 10 times runs :829ms

cutoff : 560000 Time required for 10 times runs :823ms

cutoff : 570000 Time required for 10 times runs :840ms

cutoff : 580000 Time required for 10 times runs :832ms

cutoff : 590000 Time required for 10 times runs :837ms

cutoff : 600000 Time required for 10 times runs :835ms

Thread count : 32

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :841ms

cutoff : 520000 Time required for 10 times runs :835ms

cutoff : 530000 Time required for 10 times runs :832ms

cutoff : 540000 Time required for 10 times runs :870ms

cutoff : 550000 Time required for 10 times runs :832ms

cutoff : 560000 Time required for 10 times runs :830ms

cutoff : 570000 Time required for 10 times runs :839ms

cutoff : 580000 Time required for 10 times runs :835ms

cutoff : 590000 Time required for 10 times runs :830ms

cutoff : 600000 Time required for 10 times runs :837ms

Thread count : 64

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :856ms  
cutoff : 520000 Time required for 10 times runs :832ms  
cutoff : 530000 Time required for 10 times runs :835ms  
cutoff : 540000 Time required for 10 times runs :858ms  
cutoff : 550000 Time required for 10 times runs :839ms  
cutoff : 560000 Time required for 10 times runs :839ms  
cutoff : 570000 Time required for 10 times runs :839ms  
cutoff : 580000 Time required for 10 times runs :832ms  
cutoff : 590000 Time required for 10 times runs :828ms  
cutoff : 600000 Time required for 10 times runs :840ms  
Thread count : 128

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :887ms  
cutoff : 520000 Time required for 10 times runs :858ms  
cutoff : 530000 Time required for 10 times runs :833ms  
cutoff : 540000 Time required for 10 times runs :860ms  
cutoff : 550000 Time required for 10 times runs :833ms  
cutoff : 560000 Time required for 10 times runs :847ms  
cutoff : 570000 Time required for 10 times runs :840ms  
cutoff : 580000 Time required for 10 times runs :838ms  
cutoff : 590000 Time required for 10 times runs :826ms  
cutoff : 600000 Time required for 10 times runs :830ms

---

Size of array : 4000000

Thread count : 2

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :2964ms  
cutoff : 520000 Time required for 10 times runs :2304ms  
cutoff : 530000 Time required for 10 times runs :2373ms  
cutoff : 540000 Time required for 10 times runs :2499ms

cutoff : 550000 Time required for 10 times runs :2422ms

cutoff : 560000 Time required for 10 times runs :2544ms

cutoff : 570000 Time required for 10 times runs :2337ms

cutoff : 580000 Time required for 10 times runs :2407ms

cutoff : 590000 Time required for 10 times runs :2270ms

cutoff : 600000 Time required for 10 times runs :2258ms

Thread count : 4

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :2422ms

cutoff : 520000 Time required for 10 times runs :2367ms

cutoff : 530000 Time required for 10 times runs :2238ms

cutoff : 540000 Time required for 10 times runs :2336ms

cutoff : 550000 Time required for 10 times runs :2263ms

cutoff : 560000 Time required for 10 times runs :2248ms

cutoff : 570000 Time required for 10 times runs :2299ms

cutoff : 580000 Time required for 10 times runs :2202ms

cutoff : 590000 Time required for 10 times runs :2528ms

cutoff : 600000 Time required for 10 times runs :2529ms

Thread count : 8

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1993ms

cutoff : 520000 Time required for 10 times runs :2015ms

cutoff : 530000 Time required for 10 times runs :2011ms

cutoff : 540000 Time required for 10 times runs :2005ms

cutoff : 550000 Time required for 10 times runs :2006ms

cutoff : 560000 Time required for 10 times runs :1873ms

cutoff : 570000 Time required for 10 times runs :1974ms

cutoff : 580000 Time required for 10 times runs :1985ms

cutoff : 590000 Time required for 10 times runs :1852ms

cutoff : 600000 Time required for 10 times runs :1972ms

Thread count : 16

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1412ms

cutoff : 520000 Time required for 10 times runs :1384ms

cutoff : 530000 Time required for 10 times runs :1380ms

cutoff : 540000 Time required for 10 times runs :1340ms

cutoff : 550000 Time required for 10 times runs :1348ms

cutoff : 560000 Time required for 10 times runs :1370ms

cutoff : 570000 Time required for 10 times runs :1349ms

cutoff : 580000 Time required for 10 times runs :1358ms

cutoff : 590000 Time required for 10 times runs :1346ms

cutoff : 600000 Time required for 10 times runs :1354ms

Thread count : 32

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1345ms

cutoff : 520000 Time required for 10 times runs :1356ms

cutoff : 530000 Time required for 10 times runs :1381ms

cutoff : 540000 Time required for 10 times runs :1347ms

cutoff : 550000 Time required for 10 times runs :1360ms

cutoff : 560000 Time required for 10 times runs :1352ms

cutoff : 570000 Time required for 10 times runs :1334ms

cutoff : 580000 Time required for 10 times runs :1354ms

cutoff : 590000 Time required for 10 times runs :1367ms

cutoff : 600000 Time required for 10 times runs :1362ms

Thread count : 64

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1386ms

cutoff : 520000 Time required for 10 times runs :1354ms

cutoff : 530000 Time required for 10 times runs :1377ms

cutoff : 540000 Time required for 10 times runs :1356ms

cutoff : 550000 Time required for 10 times runs :1447ms

cutoff : 560000 Time required for 10 times runs :1362ms

cutoff : 570000 Time required for 10 times runs :1342ms

cutoff : 580000 Time required for 10 times runs :1367ms

cutoff : 590000 Time required for 10 times runs :1340ms

cutoff : 600000 Time required for 10 times runs :1360ms

Thread count : 128

Degree of parallelism: 7

cutoff : 510000 Time required for 10 times runs :1349ms

cutoff : 520000 Time required for 10 times runs :1368ms

cutoff : 530000 Time required for 10 times runs :1372ms

cutoff : 540000 Time required for 10 times runs :1342ms

cutoff : 550000 Time required for 10 times runs :1335ms

cutoff : 560000 Time required for 10 times runs :1351ms

cutoff : 570000 Time required for 10 times runs :1329ms

cutoff : 580000 Time required for 10 times runs :1344ms

cutoff : 590000 Time required for 10 times runs :1348ms

cutoff : 600000 Time required for 10 times runs :1329ms

## Deduction:

It can be concluded from the results mentioned above and below in the graph that :

- After changing the cutoff value and the number of the threads for different sizes of arrays, the number of threads bigger than 16 does not improve the performance. So optimal choice of threads is 16.
- It can be concluded that for cutoff value of 20% of the size of the array, lowest performance is achieved. Thus 20% is the optimal cutoff value.

## Charts:

The same relationship can be depicted using the



# Akshaysingh Bayes (NU ID: 002956209)

## 6205 - Program Structures and Algorithms

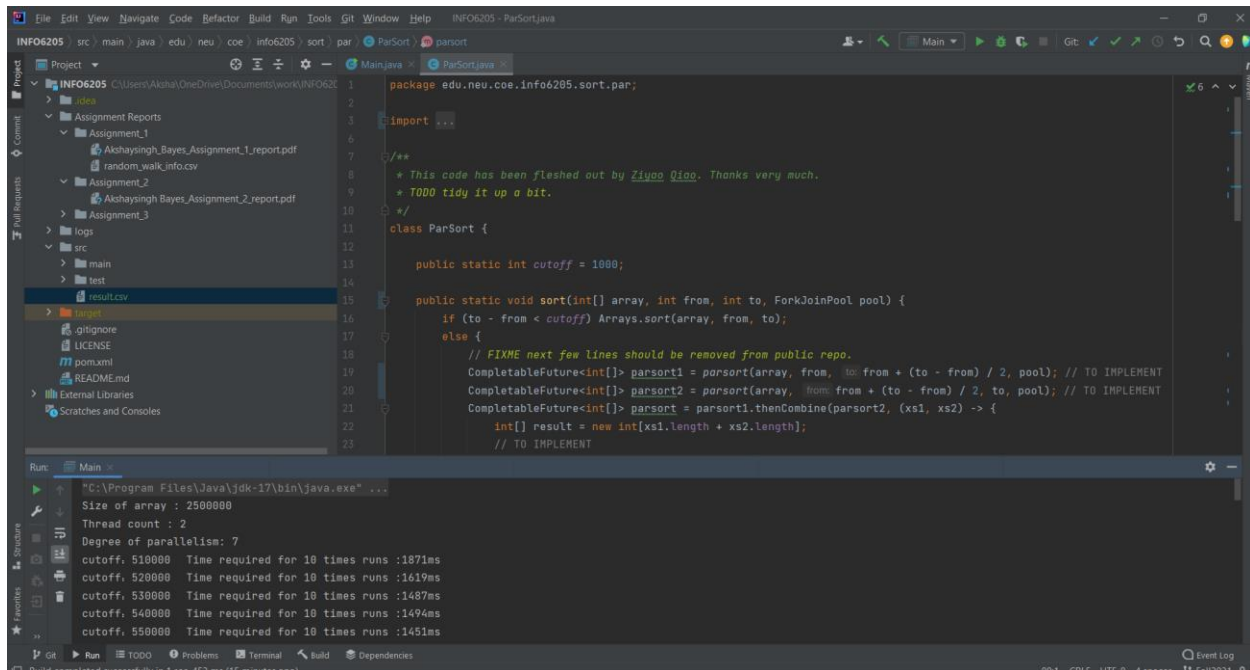
### Assignment - 5

#### Problem Statement:

To implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. Considering two different schemes for deciding whether to sort in parallel

- A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running.
- 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).

#### Output:



The screenshot shows an IDE with the following content:

```
package edu.neu.coe.info6205.sort.par;

import ...

/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * TODO tidy it up a bit.
 */
class ParSort {

    public static int cutoff = 1000;

    public static void sort(int[] array, int from, int to, ForkJoinPool pool) {
        if (to - from < cutoff) Arrays.sort(array, from, to);
        else {
            // FIXME next few lines should be removed from public repo.
            CompletableFuture<int[]> parsort1 = parsort(array, from, (to - from) / 2, pool); // TO IMPLEMENT
            CompletableFuture<int[]> parsort2 = parsort(array, (to - from) / 2, to, pool); // TO IMPLEMENT
            CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) -> {
                int[] result = new int[xs1.length + xs2.length];
                // TO IMPLEMENT
            });
        }
    }
}
```

The console output shows the following results:

```
Run: Main
C:\Program Files\Java\jdk-17\bin\java.exe* ...
Size of array : 2500000
Thread count : 2
Degree of parallelism: 7
cutoff, 510000 Time required for 10 times runs :1871ms
cutoff, 520000 Time required for 10 times runs :1619ms
cutoff, 530000 Time required for 10 times runs :1487ms
cutoff, 540000 Time required for 10 times runs :1494ms
cutoff, 550000 Time required for 10 times runs :1451ms
```

#### Console Output:

Size of array : 2500000

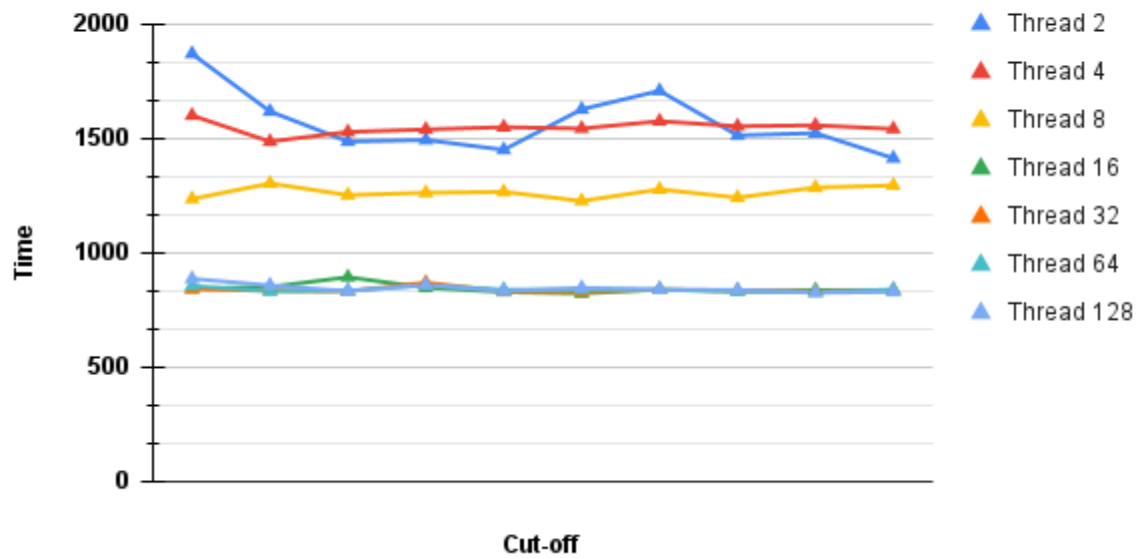
Thread count : 2

Degree of parallelism: 7

multi-line chart plotted below.

## Parallel Sorting

ArraySize = 2500000



## Parallel Sort

ArraySize = 4000000

