CS353 Operating systems Lab
Nilay Ganvit - 200001053
12th September 2022

Lab 4

Code:

```
import java.lang.System;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Random;
import java.util.*;

class MergeSort{

   private static final int MAX_THREADS = 4;

   private static class SortThreads extends Thread{
       SortThreads(Integer[] array, int begin, int end){
           super(()->{
               MergeSort.mergeSort(array, begin, end);
           });
           this.start();
       }
   }

   public static void threadedSort(Integer[] array){

       long time = System.currentTimeMillis();
       final int length = array.length;

       boolean exact = length%MAX_THREADS == 0;
       int maxlim = exact? length/MAX_THREADS: length/(MAX_THREADS-1);

       maxlim = maxlim < MAX_THREADS? MAX_THREADS : maxlim;

       final ArrayList<SortThreads> threads = new ArrayList<>();

       for(int i=0; i < length; i+=maxlim){
           int beg = i;
```

```
            int remain = (length)-i;
            int end = remain < maxlim? i+(remain-1): i+(maxlim-1);
            final SortThreads t = new SortThreads(array, beg, end);

            threads.add(t);
        }
        for(Thread t: threads){
            try{

                t.join();
            } catch(InterruptedException ignored){}
        }

        for(int i=0; i < length; i+=maxlim){
            int mid = i == 0? 0 : i-1;
            int remain = (length)-i;
            int end = remain < maxlim? i+(remain-1): i+(maxlim-1);

            merge(array, 0, mid, end);
        }
        time = System.currentTimeMillis() - time;
        System.out.println("Time taken by Multi-Threaded Merge Sort: "+
time+ "ms");
    }

    static void swap(int[] arr, int i, int j)
    {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
    }

    static int partition(int[] arr, int low, int high)
    {

    int pivot = arr[high];

    int i = (low - 1);
     for(int j = low; j <= high - 1; j++)
    {
```

```
        if (arr[j] < pivot)
        {
            i++;
            swap(arr, i, j);
        }
    }
    swap(arr, i + 1, high);
    return (i + 1);
    }

    static void quickSort(int[] arr, int low, int high)
    {
        if (low < high)
        {

            int pi = partition(arr, low, high);

            quickSort(arr, low, pi - 1);
            quickSort(arr, pi + 1, high);
        }
    }

    public static void mergeSort(Integer[] array, int begin, int end){
        if (begin<end){
            int mid = (begin+end)/2;
            quickSort(array, begin, mid);
            quickSort(array, mid+1, end);
            merge(array, begin, mid, end);
        }
    }

    public static void merge(Integer[] array, int begin, int mid, int end){
        Integer[] temp = new Integer[(end-begin)+1];

        int i = begin, j = mid+1;
        int k = 0;

        while(i<=mid && j<=end){
            if (array[i] <= array[j]){
                temp[k] = array[i];
```

```
                i+=1;
            }else{
                temp[k] = array[j];
                j+=1;
            }
            k+=1;
        }

        while(i<=mid){
            temp[k] = array[i];
            i+=1; k+=1;
        }

        while(j<=end){
            temp[k] = array[j];
            j+=1; k+=1;
        }

        for(i=begin, k=0; i<=end; i++,k++){
            array[i] = temp[k];
        }
    }
}

class Driver{

    public static void main(String[] args){
        Scanner sc= new Scanner(System.in);
        System.out.print("Enter the number of elements:");
        int size=sc.nextInt();
        Integer list[] = new Integer[size];
        Random random = new Random();
        for(int i=0; i<size; i++){

            list[i] = random.nextInt(size+(size-1))-(size-1);
        }
    System.out.print("Generated Random Numbers = ");
    for (Integer each: list)
        System.out.print(each+", ");
    System.out.print(" \n");
```

```
    Integer[] arr = Arrays.copyOf(list, list.length);
    MergeSort.threadedSort(arr);
    System.out.print("Sorted Array= ");
    for (Integer each: arr)
        System.out.print(each+", ");
    System.out.print("\n");
    }
}
```

Input/Output In terminal:

```
nilay@Nilay-PC:~/cs353$ javac QuickSortMutliThreading.java
nilay@Nilay-PC:~/cs353$ java QuickSortMutliThreading
Enter the number of elements:10
Randomly Generated array:
0 4 3 8 1 -4 9 2 5 -2
Sortedd Array:
-4 -2 0 1 2 3 4 5 8

nilay@Nilay-PC:~/cs353$
```