Lab 6: External merge sort

Professor: Ronaldo Menezes

TA: Ivan Bogun

Department of Computer Science
Florida Institute of Technology

October 8, 2014

1 General description

In this lab you will be given a file with integers. Your task is to sort it and save a sorted file on the hard drive. As simple as that. Unfortunately the file will be too big to fit in the RAM, thus conventional methods won't work. This problem is known as External sorting[1], since eventually you will need to use external memory .

1.1 Algorithm

This algorithm is known as a 2-way external merge sort. Let S be the whole set of integers in the file, then :

1) Divide S into chunks of size T (will be a parameter in the algorithm). Sort each chunk using merge sort and save every chunk into a separate file. (You are required to implement your own merge sort)

2)Merge every two consecutive sorted files using merge procedure from the merge sort algorithm. You will have to modify it so it can sort integers from the two files without creating arrays. The result of this merging should be saved into a file. Continue until you merge every pair of consecutive files. Now you decreased the number of files by the factor of two. Continue merging rounds until only one file is left, which is supposed to be a sorted version of the input. Also, your program has to delete all files, but last one.

1.2 Useful tips

Test every function you write on the small testcase. If you want to test if your program solves the problem correctly (sorts the input file) you can do the following:

1) Write a small program which reads input file and loops through lines of the file checking if the next integer is higher or equal than the previous one

2) Check the space (physical memory) of the input and output files. Since your program has to sort, both files should be of the same size.

2 Implementation

Implement class *ExternalSort.java* [2].

```
// ExternalSort.java
public class ExternalSort {

    private int chunkSize ;            // size of the chunks
```

[1] http://en.wikipedia.org/wiki/External_sorting
[2] ExternalSort.java

```
    private String directory;           // directory where temporary files will
        be saved to

   public ExternalSort(String directory_, int chunkSize_) {
       // implement constructor
   }

       public int[] mergeSort(int[] a){
       // regular merge sort
       }

     public void completeSort(String inputFile,String outputFile) {
     // inputFile - filename of the input file
     // outputFile - filename of the output file ( new file which is a sorted
         version of the input)
     // implement external 2-way merge sort
   }
}
```

3 Optional problems

- Investigate which chunk size, T, will be optimal. Although you can find it analytically, you can run a few experiments and make a plot with running time vs chunk size to determine the optimal value.

4 Sample input-output

4.1 Input

Use the following main for testing.

```
// main method
public static void main(String[] args) throws FileNotFoundException {
ExternalSort externalsort=new
ExternalSort("C:\\Users\\TEMP\\Downloads\\temp\\",10000);
externalsort.completeSort("C:\\Users\\TEMP\\Downloads\\input\\testcase2.txt",
"C:\\Users\\TEMP\\Downloads\\output\\output2.txt");
}
```

4.2 Output

File 'output2.txt' has to be sorted. All temporary files have to be deleted.

5 Grade breakdown

basis	grade
Implementation	(60)
merge sort	20
external sort	40
Comments	(20)
General	10
Javadocs	10
Overall	(20)
Compiled	5
Style	5
Runtime	10
Total	100