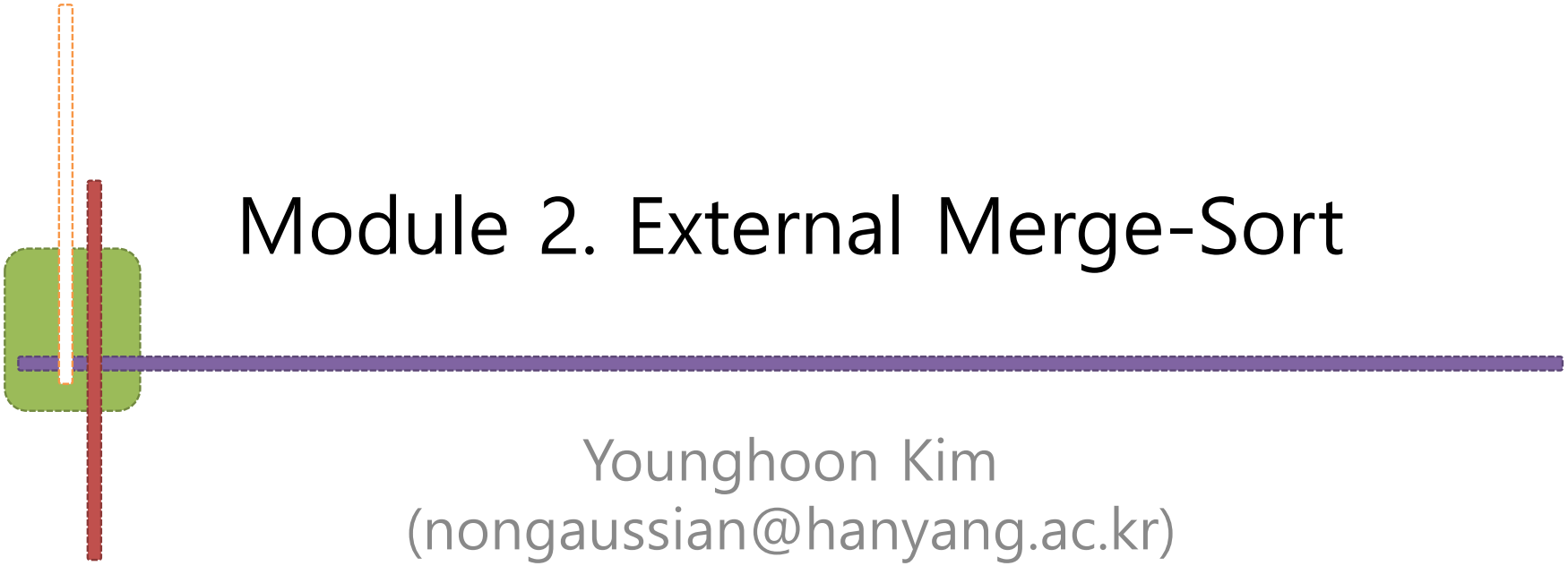

Module 2. External Merge-Sort

Younghoon Kim
(nongaussian@hanyang.ac.kr)

Goal

- Given
  - A file
    - Containing a list of triples with 3 integers (e.g., <5, 1, 2>)
    - For triples, use `org.apache.commons.lang3.tuple.MutableTriple`
- Return
  - A file
    - A list of triples sorted in the ascending order by using external merge sort
  - Sorting criteria
    - Primarily, sort by the first value
    - With tuples with an identical first value, use the second value
    - With tuples with identical first and second values, use the third value
  - Example

(4,8,4)(4,5,4)(7,9,6)(0,6,5)(6,0,3)(0,5,3)(3,1,7)(5,4,9)(4,6,6)(9,1,1)

(0,5,3)(0,6,5)(3,1,7)(4,5,4)(4,6,6)(4,8,4)(5,4,9)(6,0,3)(7,9,6)(9,1,1)

Interface

```
public interface ExternalSort {

    /**
     * Sorting the given input file to an output file
     *
     * @param infileInput file
     * @param outfileOutput file
     * @param tmpdirTemporary directory to be used for saving intermediate results on
     * @param blocksizeAvailable blocksize in the main memory of the current system
     * @param nblocksAvailable block numbers in the main memory of the current system
     * @throws IOExceptionException while performing external sort
     */
    void sort(String infile, String outfile, String tmpdir,
              int blocksize, int nblocks) throws IOException;
}
```

Code Template

- HanyangSE-submit
  - contains
    - Template codes
      (`edu.hanyang.submit.HanyangSEExternalSort.java`)
    - JUnit test codes

Complete Interface in HanyangSE-submit

- Step 1. Write your module
  - Complete edu.hanyang.submit.HanyangSEExternalSort
- Step 2. Comment out @Ignore annotation
- Step 3. Test and build your codes
  - Run “mvn test”
- Step 4. Submit your project in a zip file
  - All files with directories

Step 2. Set @Ignore annotation

- Comment out @Ignore for ExternalSortTest class

ExternalSortTest.java

```
1  package edu.hanyang;
2
3  import static org.junit.Assert.*;
16
17 //@Ignore("Delete this line to unit test stage 2")
18 public class ExternalSortTest {
19     @Before
20     public void init() {
21         clean("./tmp");
22         File resultFile = new File("./sorted.data");
23         if(resultFile.exists()) {
24             resultFile.delete();
25         }
26     }
27
```

- Set on @Ignore for the other unit test classes

*TokenizerTest.java | ExternalSortTest.java

```
1  package edu.hanyang;
2
3  import static org.junit.Assert.assertTrue;
15
16 @Ignore("Delete this line to unit test stage 1")
17 public class TokenizerTest {
18     static List<String[]> results;
19     static List<String> testSentences;
20
21     @BeforeClass
22     public static void init() {
23         results = new ArrayList<String[]>();
24
```

Partial Code from TA's Solution

```
import org.apache.commons.lang3.tuple.MutableTriple;

public void sort(String infile, String outfile, String tmpdir, int blocksize, int nblocks) throws IOException {
        1) initial phase
        ArrayList<MutableTriple<Integer, Integer, Integer>> dataArr = new ArrayList<>(nElement);
        ...

        2) n-way merge
         _externalMergeSort(tmpdir, outfile, 0);
}

private void _externalMergeSort(String tmpDir, String outputFile, int step) throws IOException {
         File[] fileArr = (new File(tmpDir + File.separator + String.valueOf(prevStep))).listFiles();
         if (fileArr.length <= nblocks - 1) {
                 for (File f : fileArr) {
                         DataInputStream dos = new ... (f.getAbsolutePath(), blocksize);
                 ...
        }
        else {
                 for (File f : fileArr) {
                         ...
                         cnt++;
                         if (cnt == nblocks - 1) {
                                 n_way_merge(...);
                 }
                 _externalMergeSort(tmpDir, outputFile, step+1);
}
```

```
public void n_way_merge(List<DataInputStream> files, String outputFile) throws IOException {
        PriorityQueue<DataManager> queue = new PriorityQueue<>
                                        (files.size(), new Comparator<DataManager>() {
                public int compare(DataManager o1, DataManager o2) {
                        return o1.tuple.compareTo(o2.tuple);
                }
        });

        while (queue.size() != 0) {
                DataManager dm = queue.poll();
                MutableTriple<Integer, Integer, Integer> tmp = dm.getTuple();
                ...
        }
}
```