Big Data Programming Assignment 2

Varaprasad Kurra Panther ID: 002430487

Question 1:

Source Code

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.log4j.Logger;
import org.apache.log4j.Level;


public class AvgTemp
      {
       private final static Logger LOGGER =
Logger.getLogger(AvgTemp.class.getName());
            public static class AvgTempMapper extends Mapper<LongWritable,
Text, Text, IntWritable >
            {
                  private static final int MISSING = 9999;
                  public void map( LongWritable key, Text value, Context
context) throws IOException, InterruptedException
                  {
                        String line = value.toString();
                        String Year = line.substring(15,19);
                        int airTemperature;
                        if(line.charAt(87)=='+')
                        {
                              airTemperature =
Integer.parseInt(line.substring(88,92));
                        }
                        else
                        {
                              airTemperature =
Integer.parseInt(line.substring(87,92));
                        }
                        String quality = line.substring(92,93);
                        if(airTemperature!=MISSING &&
quality.matches("[01459]"))
                        {
                              context.write(new Text(Year), new
IntWritable(airTemperature));
                        }
                  }
```

```
            }
            public static class AvgTempReducer extends Reducer<Text,
IntWritable, Text, FloatWritable>
            {
                  public void reduce(Text key, Iterable<IntWritable> values,
Context context) throws IOException, InterruptedException
                  {
                        int sum=0;
                        int count =0;
                        for (IntWritable value:values)
                        {
                              sum +=  value.get();
                              count +=1;
                        }
                        float avgValue = (float) (sum/(double)count);
                        context.write(key, new FloatWritable(avgValue));
                  }
            }

            public static void main(String[] args) throws Exception
              {
                Configuration conf = new Configuration();
                Job job = Job.getInstance(conf, "word count");
                Logger log = Logger.getLogger(AvgTemp.class);
                job.setJarByClass(AvgTemp.class);
                job.setMapperClass(AvgTempMapper.class);
                // job.setCombinerClass(IntSumReducer.class);
                job.setReducerClass(AvgTempReducer.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                System.exit(job.waitForCompletion(true) ? 0 : 1);

                LOGGER.log(Level.INFO, "Job name  - " + job.getJobName());
                LOGGER.log(Level.INFO, "Partitioner class  - " +
job.getPartitionerClass().getName());
                LOGGER.log(Level.INFO, "No of Reducer Tasks - " +
job.getNumReduceTasks());

                log.debug("Partitioner Class Name"+
job.getPartitionerClass().getName());
                log.debug("No Of Jobs"+ job.getCounters() );
                log.debug("The progress of the job's reduce-tasks"+
job.reduceProgress() );
              }
}
```

Command Showing starting a job

```
Activities   Terminal   Jun 16 15:02
varaprasad@varaprasad-VirtualBox: ~/output-avg-weather
jar .
varaprasad@varaprasad-VirtualBox:~$ hadoop jar AvgTemp.jar AvgTemp input-weathe
r  output-avg-weather
2020-06-16 14:57:15,652 INFO impl.MetricsConfig: loaded properties from hadoop-
metrics2.properties
2020-06-16 14:57:17,213 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot
period at 10 second(s).
2020-06-16 14:57:17,214 INFO impl.MetricsSystemImpl: JobTracker metrics system
started
2020-06-16 14:57:18,167 WARN mapreduce.JobResourceUploader: Hadoop command-line
 option parsing not performed. Implement the Tool interface and execute your ap
plication with ToolRunner to remedy this.
2020-06-16 14:57:18,611 INFO input.FileInputFormat: Total input files to proces
s : 5
2020-06-16 14:57:18,858 INFO mapreduce.JobSubmitter: number of splits:5
2020-06-16 14:57:20,305 INFO mapreduce.JobSubmitter: Submitting tokens for job:
 job_local179212206_0001
2020-06-16 14:57:20,307 INFO mapreduce.JobSubmitter: Executing with tokens: []
2020-06-16 14:57:21,310 INFO mapreduce.Job: The url to track the job: http://lo
calhost:8080/
2020-06-16 14:57:21,311 INFO mapreduce.Job: Running job: job_local179212206_000
1
2020-06-16 14:57:21,353 INFO mapred.LocalJobRunner: OutputCommitter set in conf
ig null
2020-06-16 14:57:21,489 INFO output.FileOutputCommitter: File Output Committer
Algorithm version is 2
2020-06-16 14:57:21,491 INFO output.FileOutputCommitter: FileOutputCommitter sk
ip cleanup _temporary folders under output directory:false, ignore cleanup fail
ures: false
```

Logging Information:

```
2020-06-16 15:20:05,918 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=3119285
                FILE: Number of bytes written=3369419
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
        Map-Reduce Framework
                Map input records=5456
                Map output records=5456
                Map output bytes=49104
                Map output materialized bytes=60046
                Input split bytes=590
                Combine input records=0
                Combine output records=0
                Reduce input groups=5
                Reduce shuffle bytes=60046
                Reduce input records=5456
                Reduce output records=5
                Spilled Records=10912
                Shuffled Maps =5
                Failed Shuffles=0
                Merged Map outputs=5
                GC time elapsed (ms)=699
                Total committed heap usage (bytes)=1088757760
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
```

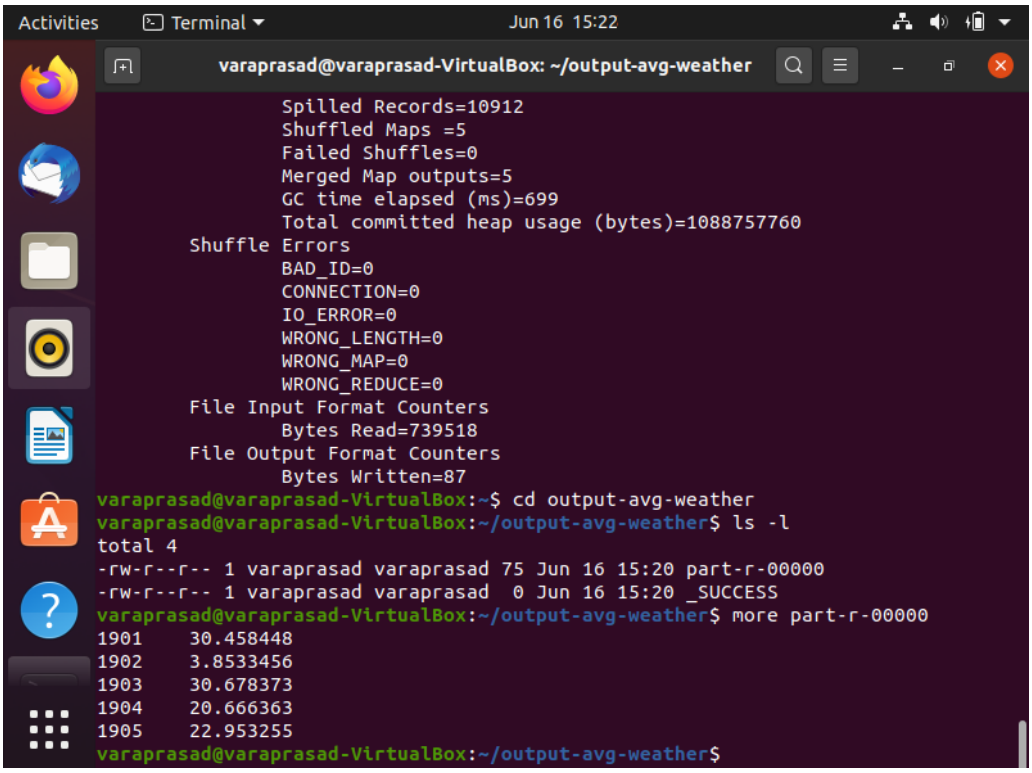
```
2020-06-16 20:54:40,024 INFO mapreduce.Job: Job job_local749669310_0001 complet
ed successfully
```

Logging Statements Used:

```
LOGGER.log(Level.INFO, "Job name  - " + job.getJobName());
LOGGER.log(Level.INFO, "Partitioner class  - " +  job.getPartitionerClass().getName());
LOGGER.log(Level.INFO, "No of Reducer Tasks - " + job.getNumReduceTasks());

log.info("Partitioner Class Name"+ job.getPartitionerClass().getName());
log.info("No Of Jobs"+ job.getCounters() );
log.info("The progress of the job's reduce-tasks"+ job.reduceProgress() );
```

Results the Avg Temperatures:



Question 2:

<u>Description about the Design:</u>

We are given Matrix (9*9 order) stored in the format (row, column, DataValue). So, let's discuss the design of the MapReduce algorithm. As we see, this matrix is 9 * 9, and we would like to break it into 3 vertical stripes (therefore each stripe is 9 * 3). The result will be 3 separate files, each file holds one stripe from the original matrix. Our main concentration is on the column, we use the column to split our given Data.

Map Task: The Map Task chooses the key value based on the column value. We assign the key as 0 if the column value matches with "[012]" with any one of the possible values. Similarly, we assign we check the remaining possible column values like

Key is 1 if the column matches the "[345]"

Key is 2 if the column matches the "[678]"

We then pass our values to the next level.

```
for each column observed:

        if(column.matches("[012]")){
                 column="0";
                }
        else if(column.matches("[345]")){
                column="1";}
        else{
                column="2";}
        context.write(new Text(column), new Text(row));
        }
```

***Reduce* Task:** In general, we use the Reduce Task to perform our aggregations etc. But in our case, we aren't performing any mathematical operations on the Data. We just write the data as it is.

```
    for each:

        write the observations received from Mapper;
```

Partition Task: This is the key task in our Map Reduce algorithm. We need to partition our data based on the column value. If the number of reducer task are zero, the partition class does nothing. And then we set next partitions based on the column key obtained from the Mapper. We set the partitions based on the remainder depending upon the number of reducer tasks.

```
if(numReduceTasks == 0)
        {
          return 0;  }
        if(col.matches("[012]"))
        {
          return 0;}
       else if(col.matches("[345]"))
        {
          return 1% numReduceTask
```

else {

return 2 % numReduceTasks;

}

Source Code:

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Partitioner;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class MatrixPartition
      {
            public static class MatrixMapper extends Mapper<LongWritable,
Text,Text, Text>
            {
                  public void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException
                  {
                    String row = value.toString();
                    String column = row.substring(3, 4);
                    if(column.matches("[012]"))
                     {
                        column="0";
                     }
                    else if(column.matches("[345]"))
                     {
                        column="1";
                     }
                    else
                     {
                        column="2";
                     }
                    context.write(new Text(column), new Text(row));
                  }
            }

         public static class MatrixReducer extends Reducer<Text, Text, Text,
Text>
         {
               public void reducer(Text key, Text value, Context context)
throws IOException, InterruptedException {
                        context.write(key, value);
               }
         }
```

```java
    public static class MatrixPartitioner extends
    Partitioner < Text, Text >
    {
       public int getPartition(Text key, Text value, int numReduceTasks)
       {
          String str = value.toString();
          String col = str.substring(3,4);

          if(numReduceTasks == 0)
          {
             return 0;
          }

          if(col.matches("[012]"))
          {
             return 0;
          }

          else if(col.matches("[345]"))
          {
             return 1% numReduceTasks;
          }
          else
          {
             return 2 % numReduceTasks;
          }

       }
    }

    public static void main(String[] args) throws Exception
       {
             Configuration conf = new Configuration();
             Job job = Job.getInstance(conf, "Matrix Partition");
             job.setJarByClass(MatrixPartition.class);
             job.setMapperClass(MatrixMapper.class);
             job.setReducerClass(MatrixReducer.class);
             job.setPartitionerClass(MatrixPartitioner.class);
             job.setOutputKeyClass(Text.class);
             job.setOutputValueClass(Text.class);
             job.setNumReduceTasks(3);
             FileInputFormat.addInputPath(job, new Path(args[0]));
             FileOutputFormat.setOutputPath(job, new Path(args[1]));
             System.exit(job.waitForCompletion(true) ? 0 : 1);
   }
}
```

Screen shot showing the command to start the job.

```
Activities   Terminal   Jun 16 14:43
varaprasad@varaprasad-VirtualBox: ~/output-matrix
-rwxrwxr-x 1 varaprasad varaprasad      4846 Jun 14 14:35 TablePartition.jar
drwxr-xr-x 2 varaprasad varaprasad      4096 Jun  9 23:03 Templates
drwxr-xr-x 2 varaprasad varaprasad      4096 Jun  9 23:03 Videos
varaprasad@varaprasad-VirtualBox:~$ hadoop jar MatrixPartition.jar MatrixPartit
ion input-matrix  output-matrix
2020-06-16 14:40:52,428 INFO impl.MetricsConfig: loaded properties from hadoop-
metrics2.properties
2020-06-16 14:40:53,255 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot
period at 10 second(s).
2020-06-16 14:40:53,255 INFO impl.MetricsSystemImpl: JobTracker metrics system
started
2020-06-16 14:40:53,690 WARN mapreduce.JobResourceUploader: Hadoop command-line
 option parsing not performed. Implement the Tool interface and execute your ap
plication with ToolRunner to remedy this.
2020-06-16 14:40:54,022 INFO input.FileInputFormat: Total input files to proces
s : 1
2020-06-16 14:40:54,219 INFO mapreduce.JobSubmitter: number of splits:1
2020-06-16 14:40:55,448 INFO mapreduce.JobSubmitter: Submitting tokens for job:
 job_local1158547569_0001
2020-06-16 14:40:55,453 INFO mapreduce.JobSubmitter: Executing with tokens: []
2020-06-16 14:40:56,329 INFO mapreduce.Job: The url to track the job: http://lo
calhost:8080/
2020-06-16 14:40:56,330 INFO mapreduce.Job: Running job: job_local1158547569_00
01
2020-06-16 14:40:56,350 INFO mapred.LocalJobRunner: OutputCommitter set in conf
ig null
2020-06-16 14:40:56,438 INFO output.FileOutputCommitter: File Output Committer
Algorithm version is 2
2020-06-16 14:40:56,443 INFO output.FileOutputCommitter: FileOutputCommitter sk
```

Screen shot showing the results:

```
Activities   Terminal   Jun 16 14:43
varaprasad@varaprasad-VirtualBox: ~/output-matrix
-rw-r--r-- 1 varaprasad varaprasad  87 Jun 16 14:40 part-r-00002
-rw-r--r-- 1 varaprasad varaprasad   0 Jun 16 14:40 _SUCCESS
varaprasad@varaprasad-VirtualBox:~/output-matrix$ more part-r-00000
0       (4,2,42)
0       (8,2,82)
0       (1,2,12)
0       (7,1,71)
0       (5,1,51)
0       (2,2,22)
0       (3,0,30)
0       (1,1,11)
0       (8,0,80)
0       (0,1,1)
varaprasad@varaprasad-VirtualBox:~/output-matrix$ more part-r-00001
1       (8,4,84)
1       (3,4,34)
1       (6,3,63)
1       (4,5,45)
1       (2,3,23)
varaprasad@varaprasad-VirtualBox:~/output-matrix$ more part-r-00002
2       (8,8,88)
2       (0,7,7)
2       (8,6,86)
2       (2,8,28)
2       (7,8,78)
2       (1,6,16)
2       (6,7,67)
2       (5,6,56)
varaprasad@varaprasad-VirtualBox:~/output-matrix$
```