

SUBJECT-BIG DATA TECHNOLOGY  
PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION

```
Map.java MatrixMultiply.java Reduce.java
1 import org.apache.hadoop.conf.*;
2 import org.apache.hadoop.io.LongWritable;
3 import org.apache.hadoop.io.Text;
4 import org.apache.hadoop.mapreduce.Mapper;
5
6 import java.io.IOException;
7
8 public class Map
9 extends org.apache.hadoop.mapreduce.Mapper<LongWritable, Text, Text, Text> {
10 @Override
11 public void map(LongWritable key, Text value, Context context)
12 throws IOException, InterruptedException {
13     Configuration conf = context.getConfiguration();
14     int m = Integer.parseInt(conf.get("m"));
15     int p = Integer.parseInt(conf.get("p"));
16     String line = value.toString();
17     // (M, i, j, Mij);
18     String[] indicesAndValue = line.split(",");
19     Text outputKey = new Text();
20     Text outputValue = new Text();
21     if (indicesAndValue[0].equals("M")) {
22         for (int k = 0; k < p; k++) {
23             outputKey.set(indicesAndValue[1] + "," + k);
24             // outputKey.set(i,k);
25             outputValue.set(indicesAndValue[0] + "," + indicesAndValue[2]
26                 + "," + indicesAndValue[3]);
27             // outputValue.set(M,j,Mij);
28             context.write(outputKey, outputValue);
29         }
30     } else {
31         // (N, j, k, Njk);
32         for (int i = 0; i < m; i++) {
33             outputKey.set(i + "," + indicesAndValue[2]);
34             outputValue.set("N," + indicesAndValue[1] + ","
35                 + indicesAndValue[3]);
36             context.write(outputKey, outputValue);
37         }
38     }
39 }
40 }
```

SUBJECT-BIG DATA TECHNOLOGY  
PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION

```
import org.apache.hadoop.io.Text;

import java.io.IOException;
import java.util.HashMap;

public class Reduce
    extends org.apache.hadoop.mapreduce.Reducer<Text, Text, Text, Text> {
    @Override
    public void reduce(Text key, Iterable<Text> values, Context context)
        throws IOException, InterruptedException {
        String[] value;
        //key=(i,k),
        //Values = [(M/N,j,V/W),...]
        HashMap<Integer, Float> hashA = new HashMap<Integer, Float>();
        HashMap<Integer, Float> hashB = new HashMap<Integer, Float>();
        for (Text val : values) {
            value = val.toString().split(",");
            if (value[0].equals("M")) {
                hashA.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
            } else {
                hashB.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
            }
        }
        int n = Integer.parseInt(context.getConfiguration().get("n"));
        float result = 0.0f;
        float m_ij;
        float n_jk;
        for (int j = 0; j < n; j++) {
            m_ij = hashA.containsKey(j) ? hashA.get(j) : 0.0f;
            n_jk = hashB.containsKey(j) ? hashB.get(j) : 0.0f;
            result += m_ij * n_jk;
        }
        if (result != 0.0f) {
            context.write(null,
                new Text(key.toString() + "," + Float.toString(result)));
        }
    }
}
```

**SUBJECT-BIG DATA TECHNOLOGY**  
**PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION**

```
import org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class MatrixMultiply {

    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: MatrixMultiply <in_dir> <out_dir>");
            System.exit(2);
        }
        Configuration conf = new Configuration();
        // M is an m-by-n matrix; N is an n-by-p matrix.
        conf.set("m", "1000");
        conf.set("n", "100");
        conf.set("p", "1000");
        @SuppressWarnings("deprecation")
        Job job = new Job(conf, "MatrixMultiply");
        job.setJarByClass(MatrixMultiply.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

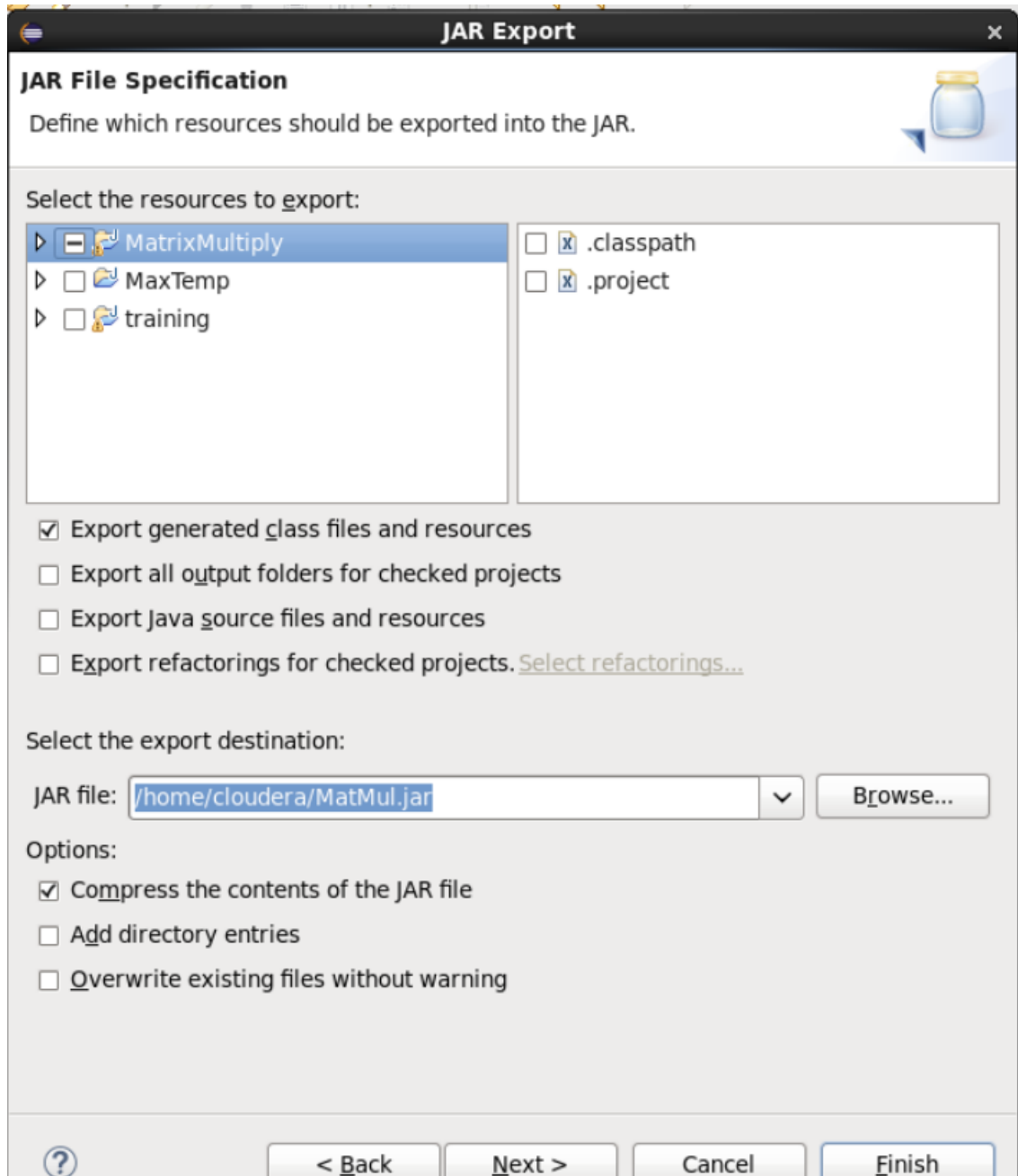
        job.setMapperClass(Map.class);
        job.setReducerClass(Reduce.class);

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        job.waitForCompletion(true);
    }
}
```

SUBJECT-BIG DATA TECHNOLOGY  
PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION



SUBJECT-BIG DATA TECHNOLOGY  
PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
[cloudera@quickstart ~]$ hdfs dfs -mkdir /matinput  
[cloudera@quickstart ~]$ hdfs dfs -put /home/cloudera/Desktop/M /matinput/  
[cloudera@quickstart ~]$ hdfs dfs -ls /matinput  
Found 1 items  
-rw-r--r-- 1 cloudera supergroup 108 2022-02-24 07:00 /matinput/M  
[cloudera@quickstart ~]$
```

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
-rw-r--r-- 1 cloudera supergroup 108 2022-02-24 07:00 /matinput/M  
[cloudera@quickstart ~]$ hdfs dfs -cat /matinput/M  
M,0,0,10  
M,0,2,9  
M,0,3,9  
M,0,5,9  
M,0,6,9  
M,0,11,8  
M,0,13,9  
M,0,16,10  
M,0,22,8  
M,0,25,10  
M,0,32,10  
M,0,34,10  
[cloudera@quickstart ~]$ hdfs dfs -cat /matinput/N  
cat: '/matinput/N': No such file or directory  
[cloudera@quickstart ~]$ hdfs dfs -put /home/cloudera/Desktop/N /matinput/  
[cloudera@quickstart ~]$ hdfs dfs -ls /matinput  
Found 2 items  
-rw-r--r-- 1 cloudera supergroup 108 2022-02-24 07:00 /matinput/M  
-rw-r--r-- 1 cloudera supergroup 111 2022-02-24 07:04 /matinput/N  
[cloudera@quickstart ~]$ hdfs dfs -cat /matinput/N  
N,0,2,9  
N,0,7,8  
N,0,8,10  
N,0,15,8  
N,0,17,10  
N,0,20,10  
N,0,21,8  
N,0,25,8  
N,0,31,10  
N,0,32,10  
N,0,35,10  
N,0,36,8  
[cloudera@quickstart ~]$
```

**SUBJECT-BIG DATA TECHNOLOGY**  
**PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION**

```
cloudera@quickstart:~$ hadoop jar /home/cloudera/MatMul.jar MatrixMultiply /matinput/* /output_mat
22/02/24 07:28:19 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
22/02/24 07:28:23 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool i
nterface and execute your application with ToolRunner to remedy this.
22/02/24 07:28:25 INFO input.FileInputFormat: Total input paths to process : 2
22/02/24 07:28:26 INFO mapreduce.JobSubmitter: number of splits:2
22/02/24 07:28:27 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1645705606640_0001
22/02/24 07:28:29 INFO impl.YarnClientImpl: Submitted application application_1645705606640_0001
22/02/24 07:28:29 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_164
5705606640_0001/
22/02/24 07:28:29 INFO mapreduce.Job: Running job: job_1645705606640_0001
22/02/24 07:29:00 INFO mapreduce.Job: Job job_1645705606640_0001 running in uber mode : false
22/02/24 07:29:00 INFO mapreduce.Job: map 0% reduce 0%
22/02/24 07:29:36 INFO mapreduce.Job: map 100% reduce 0%
22/02/24 07:29:58 INFO mapreduce.Job: map 100% reduce 100%
22/02/24 07:29:59 INFO mapreduce.Job: Job job_1645705606640_0001 completed successfully
22/02/24 07:29:59 INFO mapreduce.Job: Counters: 49
    File System Counters
      FILE: Number of bytes read=360366
      FILE: Number of bytes written=1053206
      FILE: Number of read operations=0
      FILE: Number of large read operations=0
      FILE: Number of write operations=0
      HDFS: Number of bytes read=433
      HDFS: Number of bytes written=123
      HDFS: Number of read operations=9
      HDFS: Number of large read operations=0
      HDFS: Number of write operations=2
    Job Counters
      Launched map tasks=2
      Launched reduce tasks=1
      Data-local map tasks=2
      Total time spent by all maps in occupied slots (ms)=66511
      Total time spent by all reduces in occupied slots (ms)=17443
      Total time spent by all map tasks (ms)=66511
      Total time spent by all reduce tasks (ms)=17443
      Total vcore-seconds taken by all map tasks=66511
      Total vcore-seconds taken by all reduce tasks=17443
      Total megabyte-seconds taken by all map tasks=68107264
    Map-Reduce Framework
      Map input records=24
      Map output records=24000
      Map output bytes=312360
      Map output materialized bytes=360372
      Input split bytes=214
      Combine input records=0
      Combine output records=0
      Reduce input groups=12988
      Reduce shuffle bytes=360372
      Reduce input records=24000
      Reduce output records=12
      Spilled Records=48000
      Shuffled Maps =2
      Failed Shuffles=0
      Merged Map outputs=2
      GC time elapsed (ms)=909
      CPU time spent (ms)=8150
      Physical memory (bytes) snapshot=542846976
      Virtual memory (bytes) snapshot=4507574272
      Total committed heap usage (bytes)=391979008
    Shuffle Errors
      BAD_ID=0
      CONNECTION=0
      IO_ERROR=0
      WRONG_LENGTH=0
      WRONG_MAP=0
      WRONG_REDUCE=0
    File Input Format Counters
      Bytes Read=219
    File Output Format Counters
      Bytes Written=123
cloudera@quickstart ~$
```

**SUBJECT-BIG DATA TECHNOLOGY**

**PRACTICAL 5- MAP REDUCE MATRIX MULTIPLICATION**

```
[cloudera@quickstart ~]$ hdfs dfs -ls /output_mat
Found 2 items
-rw-r--r-- 1 cloudera supergroup 0 2022-02-24 07:29 /output_mat/_SUCCESS
-rw-r--r-- 1 cloudera supergroup 123 2022-02-24 07:29 /output_mat/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /output_mat/part-r-00000
0,15,80.0
0,17,100.0
0,2,90.0
0,20,100.0
0,21,80.0
0,25,80.0
0,31,100.0
0,32,100.0
0,35,100.0
0,36,80.0
0,7,80.0
0,8,100.0
[cloudera@quickstart ~]$
```

