LAB REPORT

Exercise-2

By

Ambuj Mishra

202116003

Big Data & Large-Scale Computing

DA-IICT, Gandhinagar

27-February-2022

Environment setup:

Before moving to the questions, we must first set up the hdfs environment to make all the nodes and resource managers running.

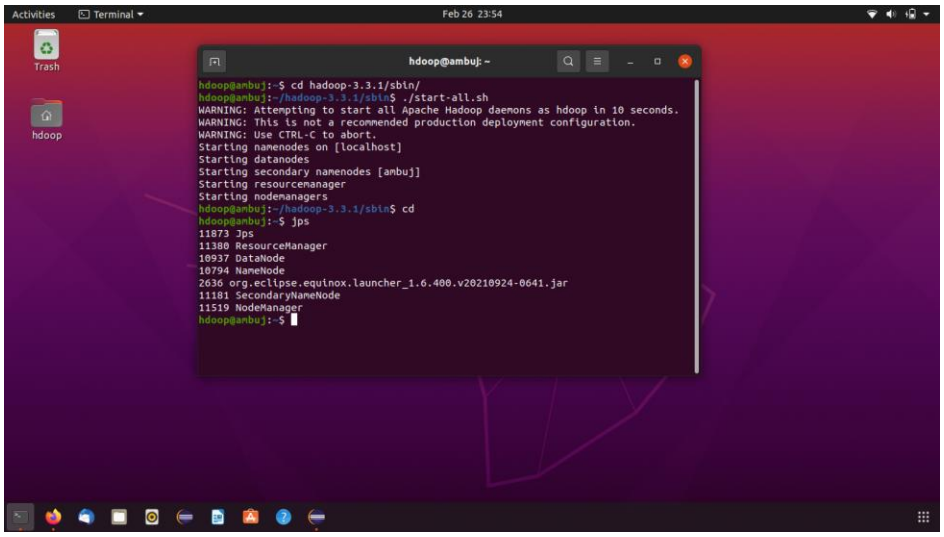


Figure 1: Starting the HADOOP environment

To run the java projects using .jar files on input data in hdfs file system, following commands are used:

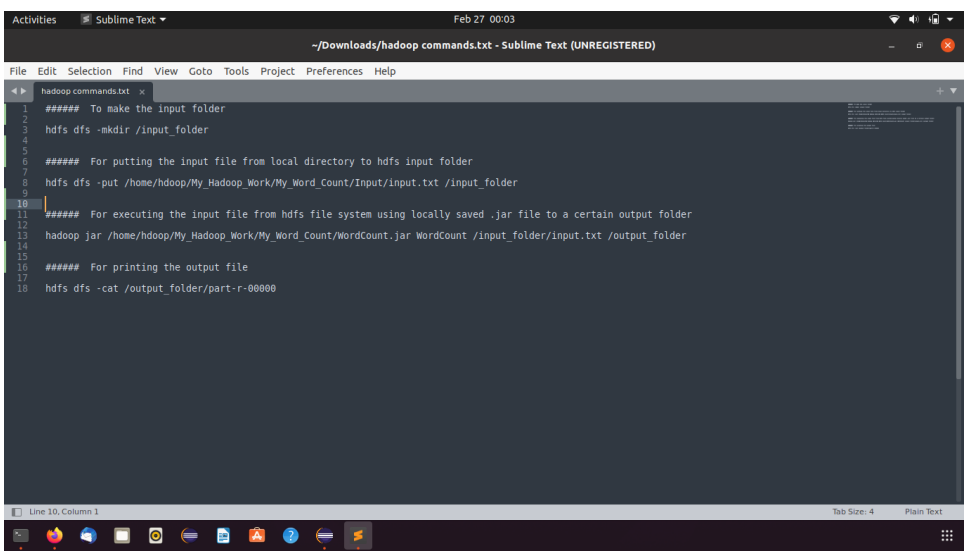


Figure 2: Commands to run a .jar file in hdfs file system

1st Question:

In the 1st question, we were asked to modify the WordCount.java file to optimize the results of the out file. We have taken the input.txt file after taking data from the white house press release on Ukraine and Russia issues. We have first run the basic MapReduce WordCount.java file on the input data to check the raw output.

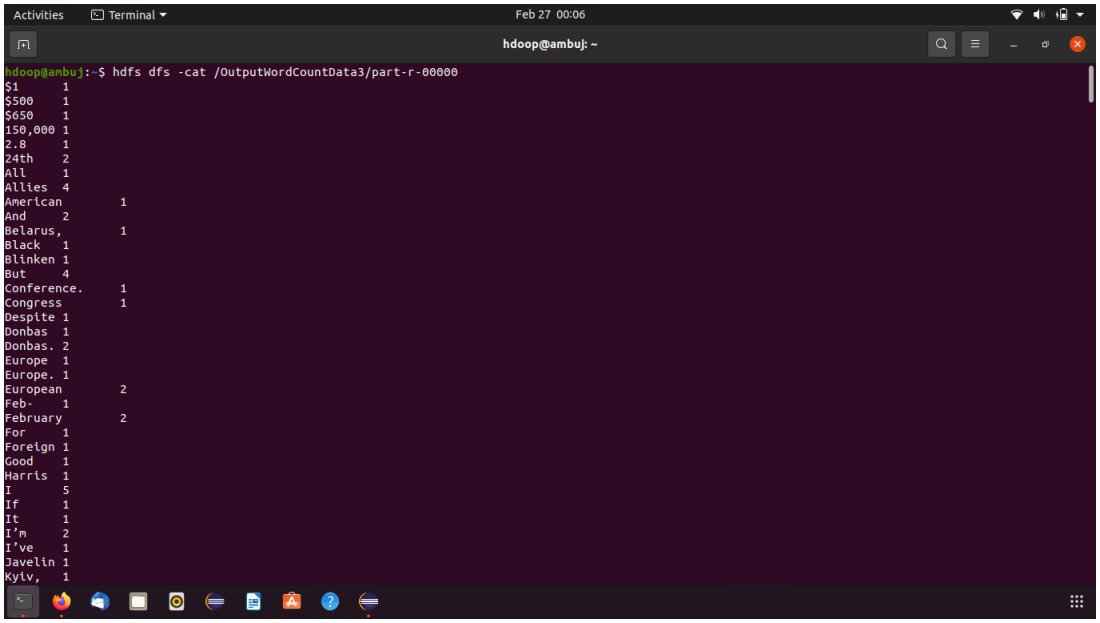


Figure 3: Running non-modified WordCount.java on input.txt

We have made the following changes to the WordCount.java code:

i. We first changed the method of tokenizing data so that we can also remove the special character symbols from the tokens.

```
StringTokenizer itr = new StringTokenizer(line, " \t\n\r\f,.-:;?![]'");
```

ii. We do not want the lower-case and upper-case words to be considered distinct words. Therefore, we have used *toLowerCase()* function.

```
word.set(itr.nextToken().toLowerCase());
```

iii. We only wanted the words that come more than 4 times in the file therefore, we have put an extra condition where the words are being recorded.

if (sum > 4) output.collect(key, new IntWritable(sum));

We have gotten the following results after running the modified WordCount.java file:

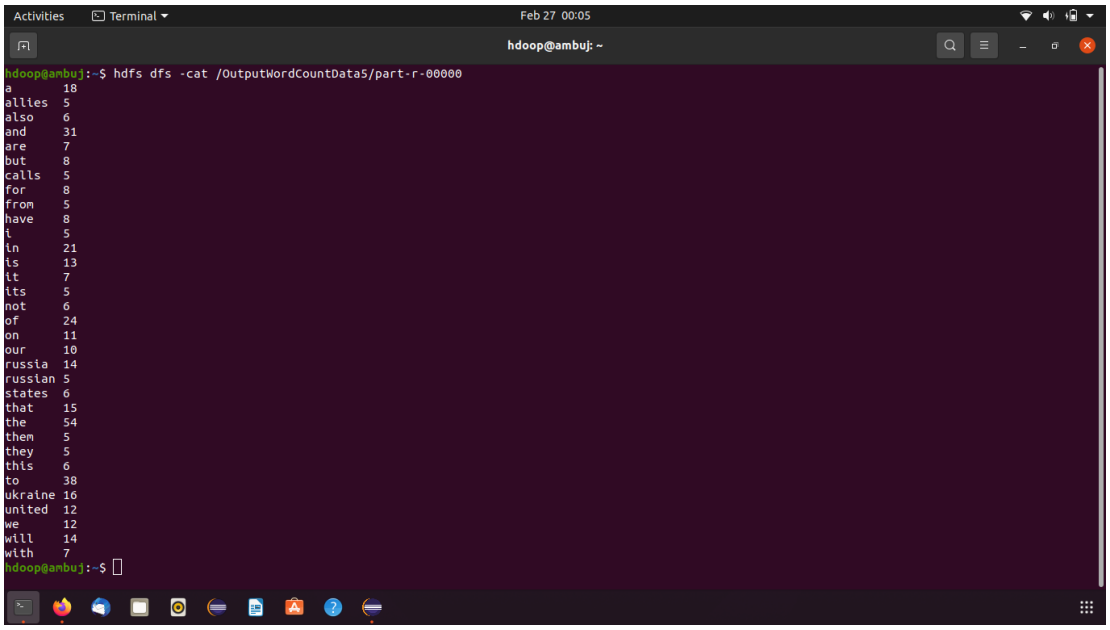


Figure 4: Running modified WordCount.java on input.txt

2nd Question:

We have used the link https://www.nber.org/research/data/us-patents to download the pairwise citation data. The dataset contains the citations and cited files. We are using the modified version of MapReduce WordCount algorithm to calculate the list of citations.

We have used MyJob.java file to calculate the list of citations. Following steps are followed to calculate the results of the list of citations:

i. We have calculated the list of citations that which files have cited the key file. We can further change the reduce function to change this into count of time the key file is cited.

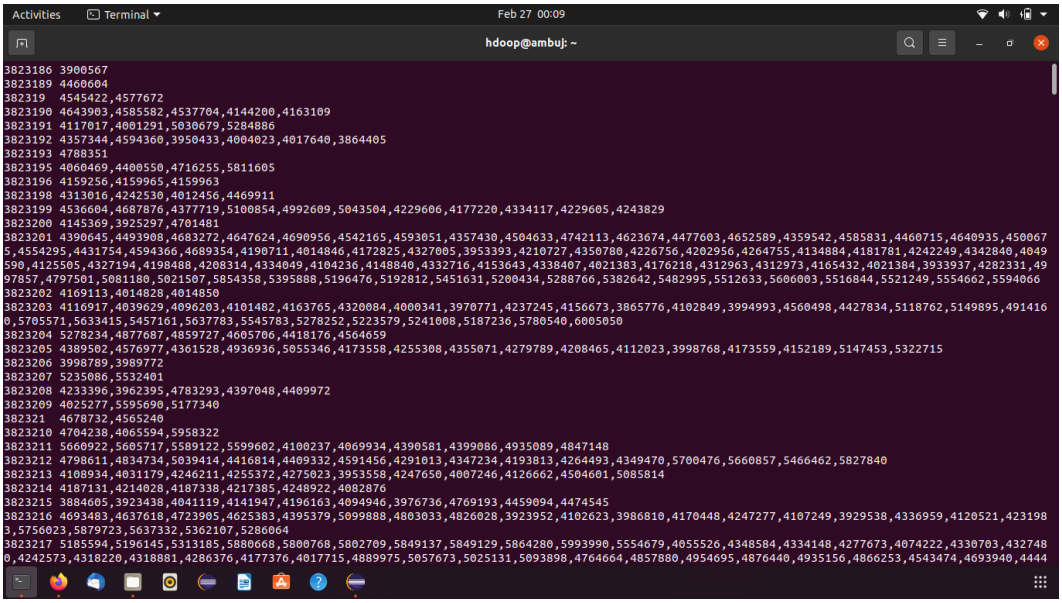
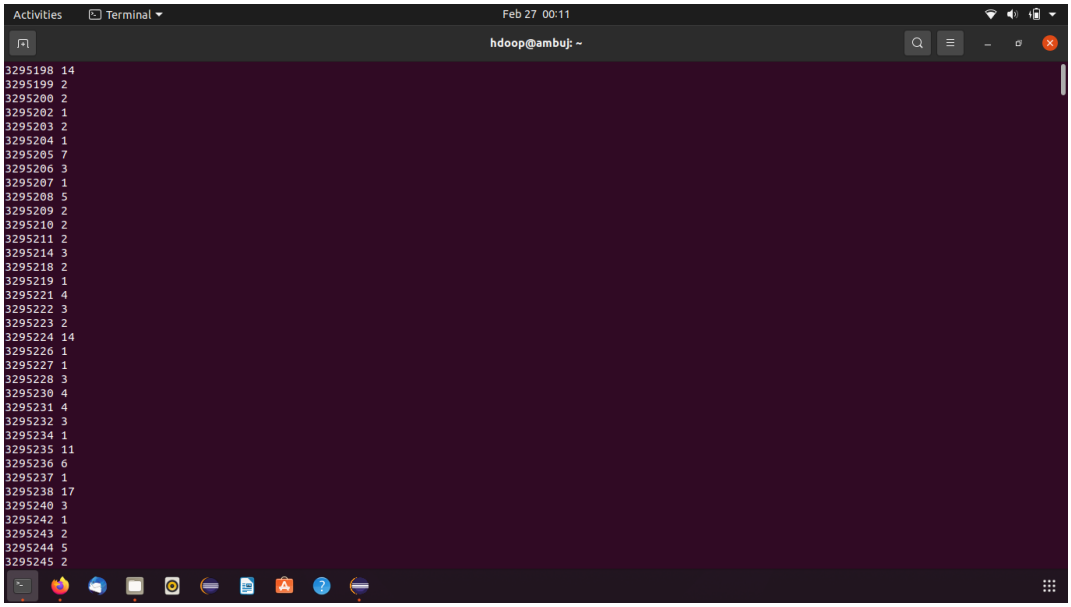


Figure 5: Non-modified MyJob.java returning list of citations

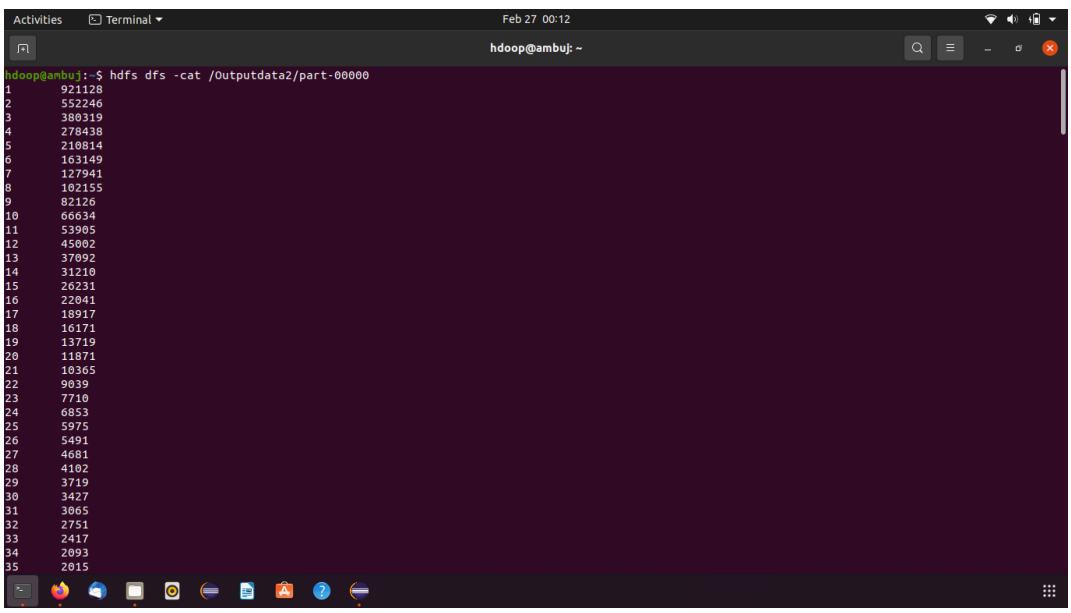
ii. We changed the reduce function to convert the list into the count of times this key file is cited.



```
3295198 14
3295199 2
3295200 2
3295202 1
3295203 2
3295204 1
3295205 7
3295206 3
3295207 1
3295208 5
3295209 2
3295210 2
3295211 2
3295214 3
3295218 2
3295219 1
3295221 4
3295222 3
3295223 2
3295224 14
3295226 1
3295227 1
3295228 3
3295230 4
3295231 4
3295232 3
3295234 1
3295235 11
3295236 6
3295237 1
3295238 17
3295240 3
3295242 1
3295243 2
3295244 5
3295245 2
```

Figure 6: Modified MyJob.java returning count of citations

iii. We have further used CitationHistogram.java file to create histogram data based on the output file that is generated after *part ii.* The output file after this step contains the number of files that have been cited once. Similarly, the number of files that have been cited twice, thrice, and so on.



```
hdoop@ambuj:~$ hdfs dfs -cat /Outputdata2/part-00000
1       921128
2       552246
3       380319
4       278438
5       210814
6       163149
7       127941
8       102155
9       82126
10      66634
11      53905
12      45002
13      37092
14      31210
15      26231
16      22041
17      18917
18      16171
19      13719
20      11871
21      10365
22      9039
23      7710
24      6853
25      5975
26      5491
27      4681
28      4102
29      3719
30      3427
31      3065
32      2751
33      2417
34      2093
35      2015
```

Figure 7: CitationHistogram.java file to create histogram data

We can also use this data to plot histogram.
