Programming Assignment-2
COEN 242, BiG Data

Anjaly George
Syeda Gousia Sultana

Due date:05/17/2020

THE PROBLEM STATEMENT:

1. Determine 100 most frequent/repeated words in the given dataset using MapReduce and Hive.
2. Determine 100 most frequent/repeated words in the given dataset considering only the words having more than 6 characters using both MapReduce and Hive.

Top 100 most frequent words using Map Reduce:

We have designed and implemented the code using default configuration vs configuration with performance tuning. This tuning has been plotted in the graphs that shows the performance improvements with respect to the change in the number of mappers and reducers used. By using the mappers we are performing parallel processing for fast execution. We are adding combiners in between Mapper and reducer, so that the reducer can have a summary information rather than the direct output from the mapper and makes the processing easier. By using the reducers we are performing the job of shuffling and sorting. These two go hand in hand for good performance. The Observations we were able to make from the code were:

1. The CPU time was significantly decreasing as we increased the number of reducers
2. GC time elapsed was increasing with the increase of number of reducers
3. Time spent in Mappers got reduced as we increased the number of reducers and the time spent in reducers increased
4. Physical memory, Virtual Memory and Heap usage was increasing with the increase in the number of reducers
5. The spilled records were fluctuating between low and high values with the change in the number of reducers

The values obtained from these observations have been recorded in the tables.

The Data used for plotting

Job Counters: The values obtained for the time spent by all maps and reduces while tuning the number of reduces.

No of reducers	Launched map tasks	Total time spent by all maps in occupied slots (Minutes)	Total time spent by all reducers in

			occupied slots(Minutes)
3	240	215.10	0.71
5	240	210.30	1.34
10	240	204.79	1.99
12	240	203.53	3.09
15	240	200.54	3.59
20	240	197.49	5.13
25	240	197.80	5.93
30	240	195.41	7.25
35	240	192.40	9.38
40	240	192.36	9.82
50	240	191.04	11.93
60	240	192.56	15.39
70	240	190.59	12.67
80	240	190.62	19.51

MR Framework: Memory and heap usage we achieve by tuning the number of reduces.

No of reducers	Physical memory(GB) snapshot	Virtual memory (GB) snapshot	Total committed heap usage (GB)
3	144.0	359.0	140.0
5	145.0	362.0	141.5
10	146.0	369.0	143.0
12	147.0	372.0	145.0

15	147.0	377.0	146.0
20	149.0	384.0	149.0
25	150.0	392.0	150.5
30	151.0	399.5	154.0
35	152.5	407.0	157.0
40	153.5	414.0	159.0
50	156.0	492.0	164.0
60	158.5	444.0	169.0
70	161.5	459.0	174.0
80	163.5	474.0	179.0

Map Reduce Execution time and Analysis

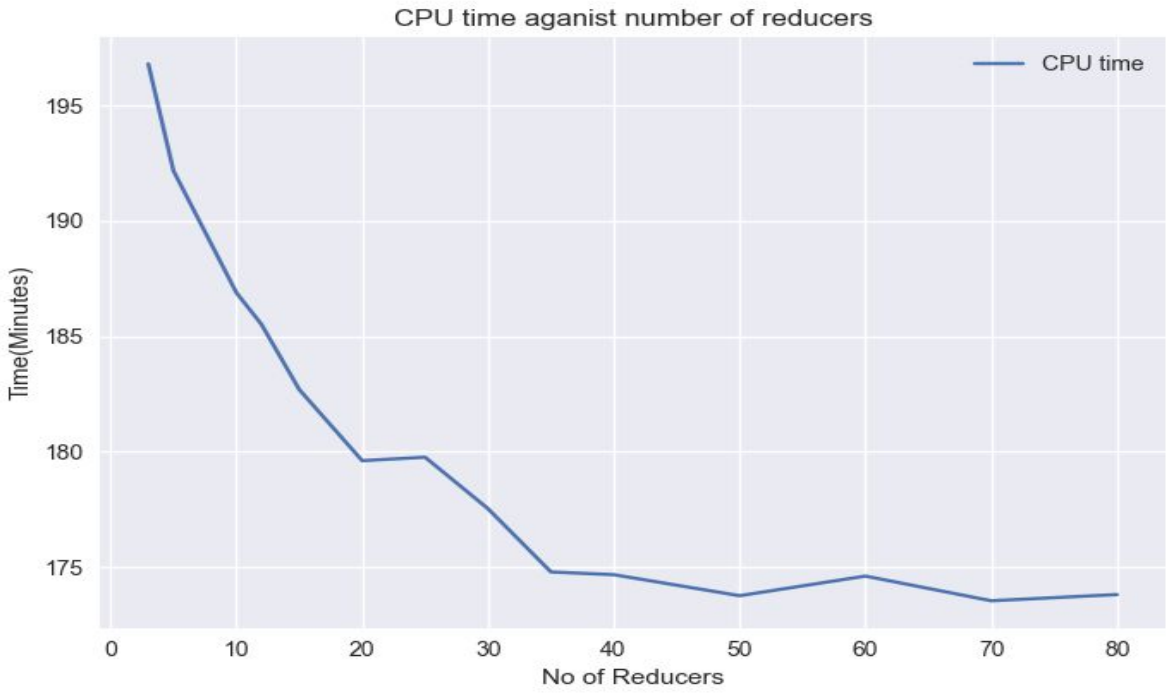
No of reducers	GC time elapsed in Minutes	CPU time spent in Minutes	Execution time in Minutes
3	1.39	196.80	2:6
5	1.36	192.17	2:3
10	1.37	186.89	2:1
12	1.38	185.51	2:1
15	1.38	182.69	1:54
20	1.41	179.61	1:56
25	1.42	179.76	1:57
30	1.40	177.53	1:56
35	1.41	174.79	1:58
40	1.41	174.67	1:57
50	1.43	173.76	1:58

60	1.44	174.61	1:54
70	1.43	173.54	1:53
80	1.47	173.81	1:48

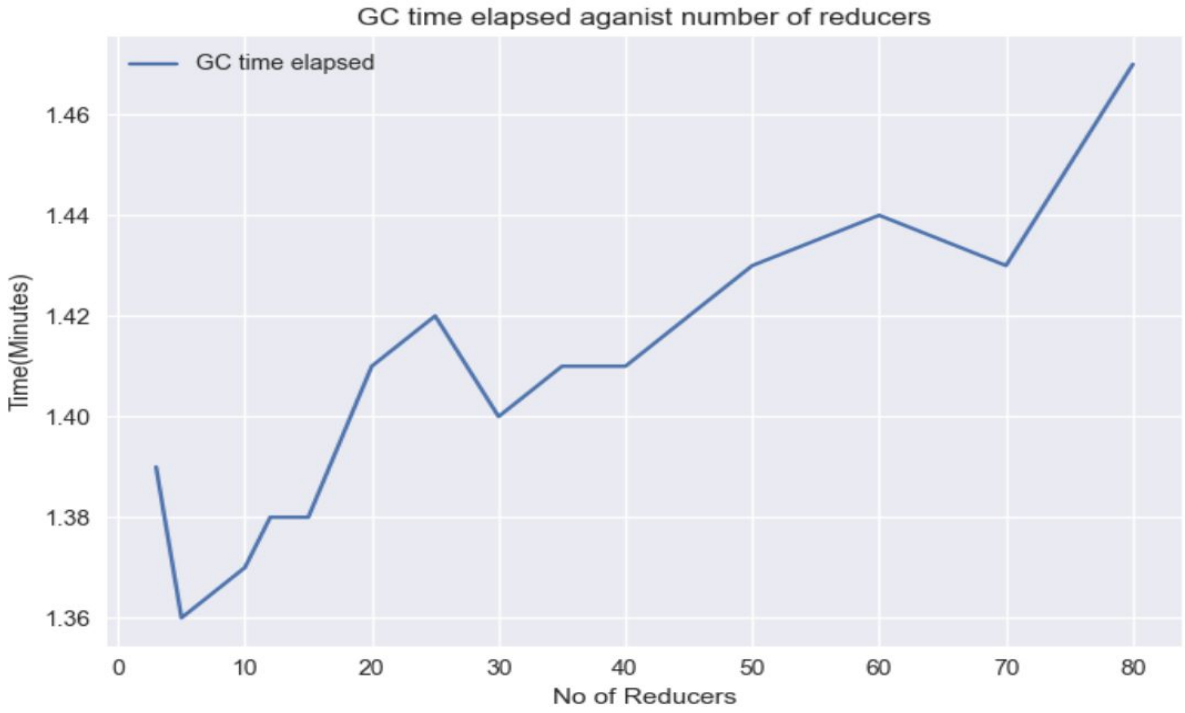
For the map reduce the following graphs have been plotted

1. CPU time vs number of reducers
2. GC time elapsed vs number of reducers
3. Heap usage, physical memory, virtual memory vs number of reducers
4. Spilled records vs number of reducers
5. Time spent in mappers vs number of reducers
6. Time spent in reducers vs number of reducers

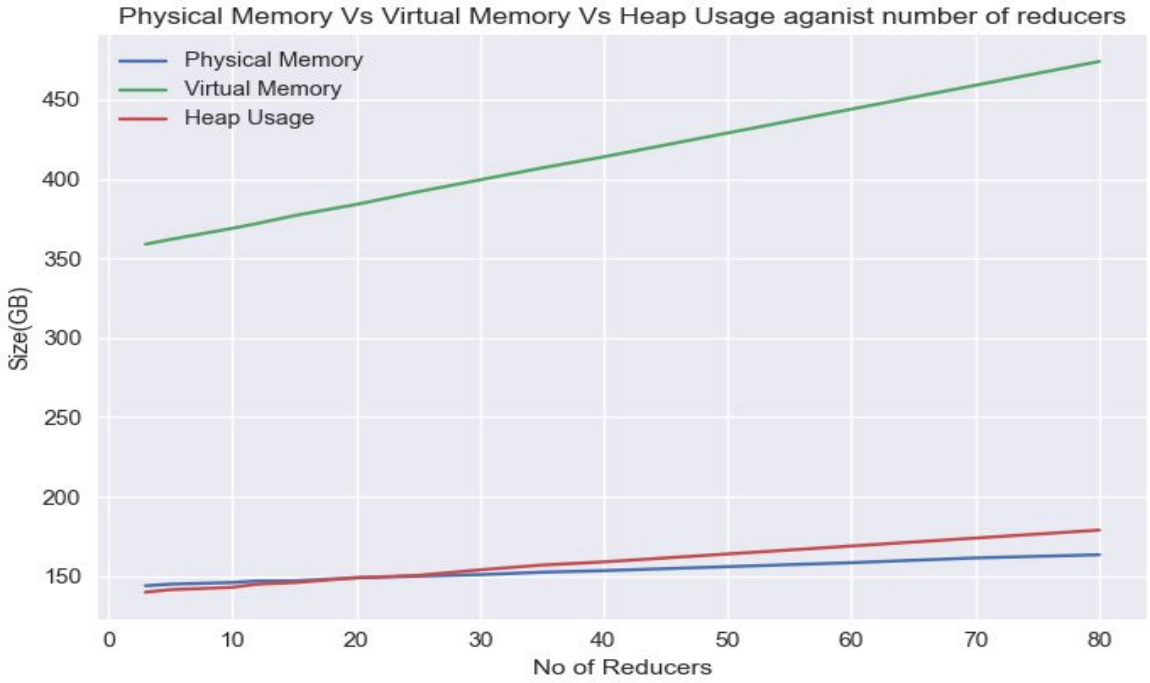
1. CPU time vs num of reducers:



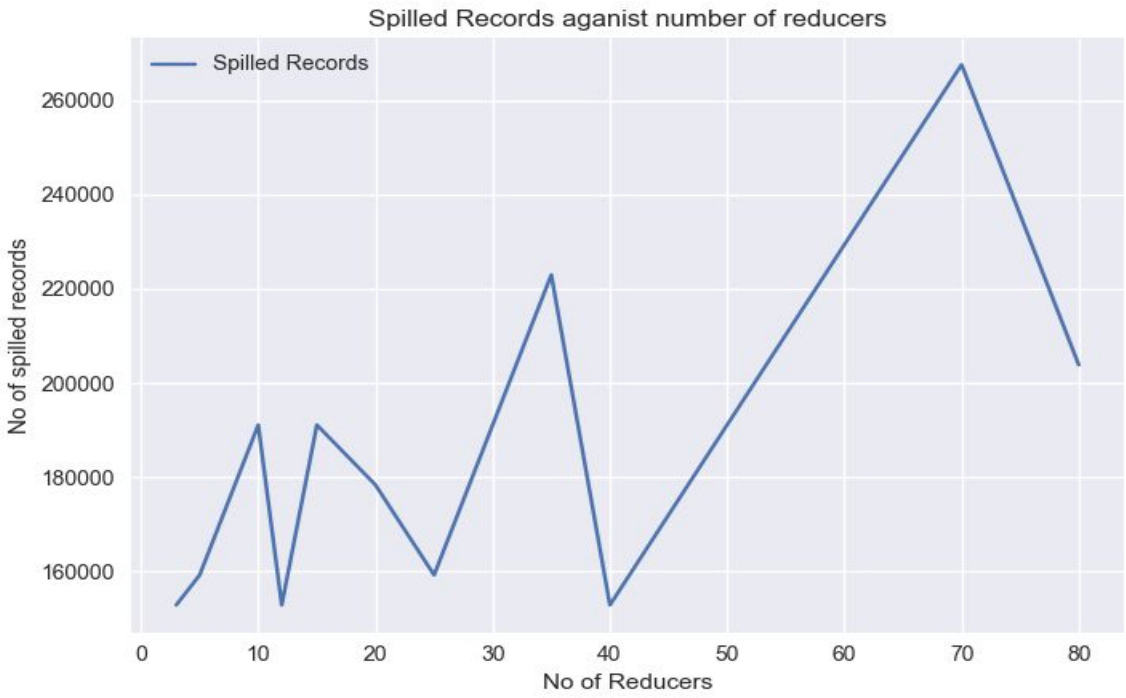
2. GC time elapsed vs number of reducers :



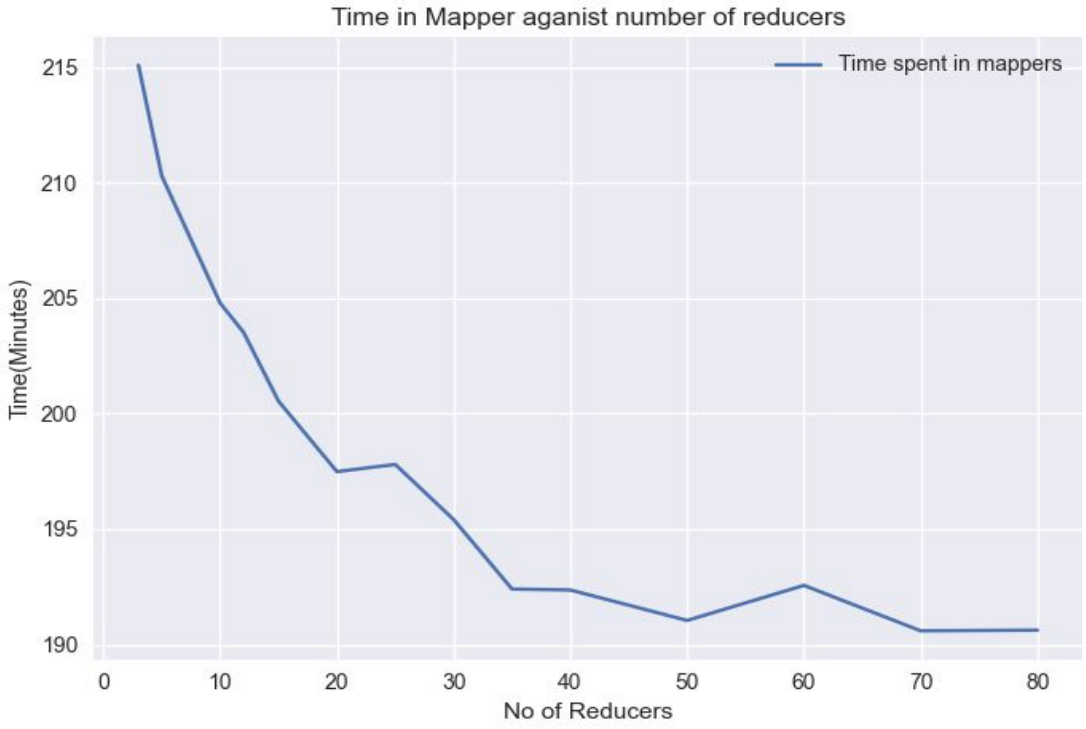
3. Heap usage, physical memory, virtual memory vs number of reducers:



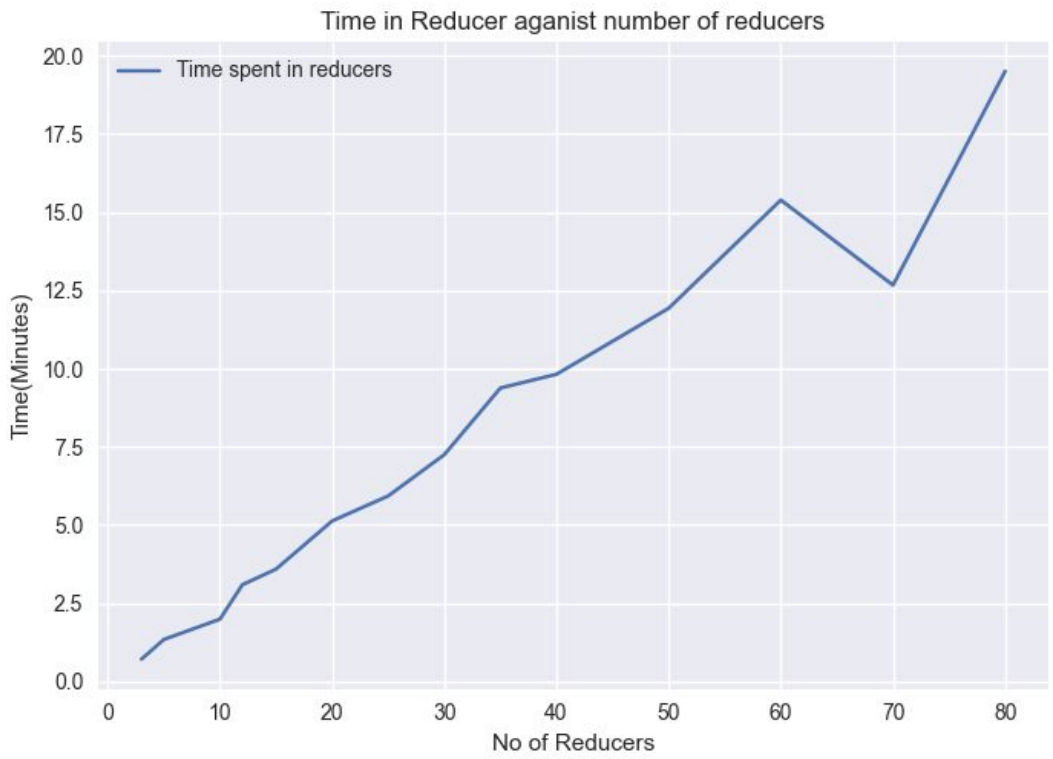
4. Spilled records vs number of reducers:



5. Time spent in mappers vs number of reducers:



6. Time spent in reducers vs number of reducers:



Top 100 most frequent words using Hive:

In Hive we have read the document and splitted the words from the document and added it to a table. Queried on the table to get the Top 100 words and Top 100 words with more than 6 characters. By increasing the number of mappers performance has been optimized. The execution time has been improved whose values have been recorded in the table. These are the following techniques we used for configuration tuning.

1. Using Compression:

By using compression the time taken is 2 min 49 sec which did not improve any performance. The number of mappers and reducers being set by default is 121 and 481 in stage-1 and 19 mappers and 1 reducer in stage-2. The following commands were used to perform compression.

set hive.exec.compress.output = true;
mapred.output.compression.codec=org.apache.hadoop.io.compress.BZip2Codec
set mapreduce.output.fileoutputformat.compress=true;

2. Without using mappers and reducers:

Time taken: 3min 09sec

These are the default mappers and reducers which got set while executing

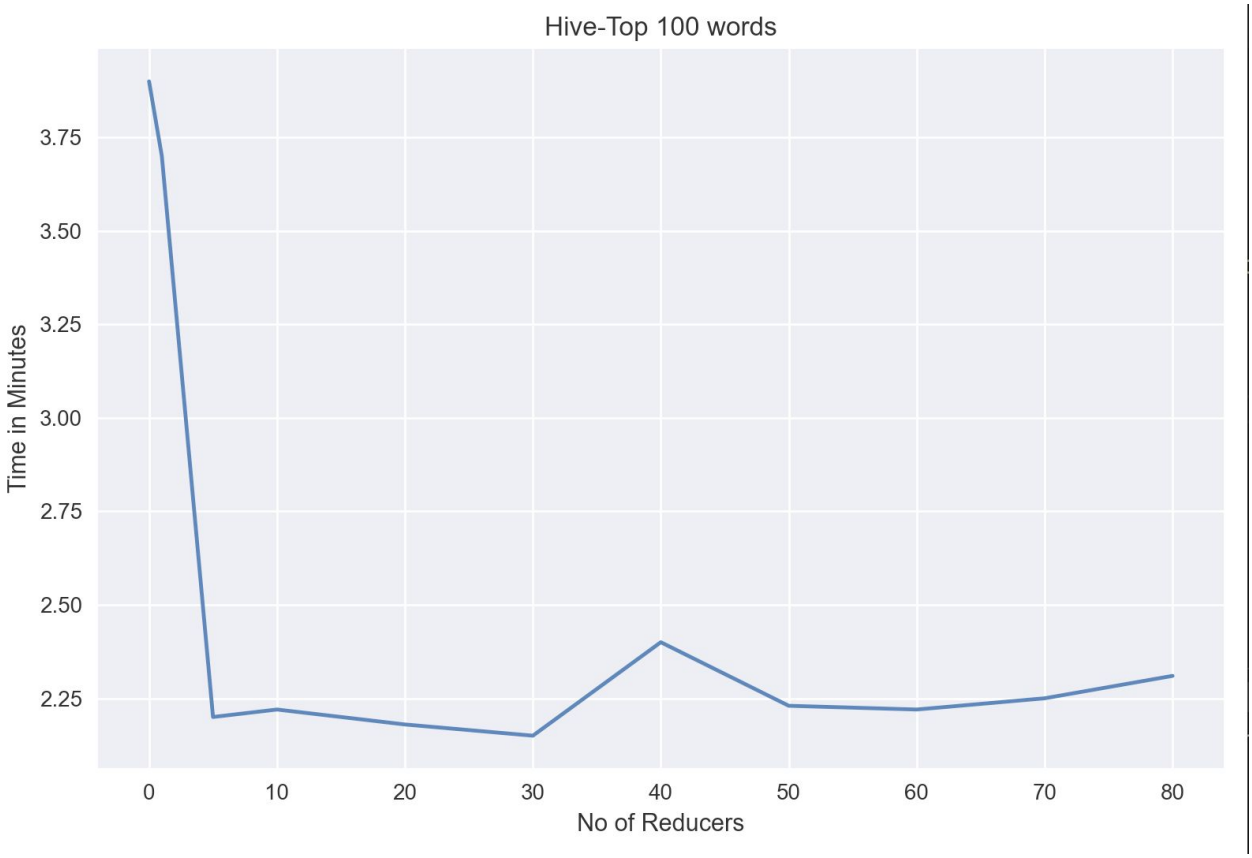
Mappers in stage: 121

Reduces in stage: 481

3. Number of Reducers vs Execution Time:

The Data used for plotting: Mappers were set by default when the reduces have been given the following constant values. The number of reduces were tuned using this command:***set mapreduce.job.reduces***

No of Reducer	Time in minutes	Default number of mappers,reducers at the time of execution Stage-1	Default number of mappers,reducers at the time of execution Stage-2
1	3.7	121,5	5,1
5	2.20	121,5	5,1
10	2.22	121,10	15,1
20	2.18	121,20	15,1
30	2.15	121,30	16,1
40	2.40	121,40	15,1
50	2.23	121,50	15,1
60	2.22	121,60	13,1
70	2.25	121,70	12,1
80	2.31	121,80	15,1



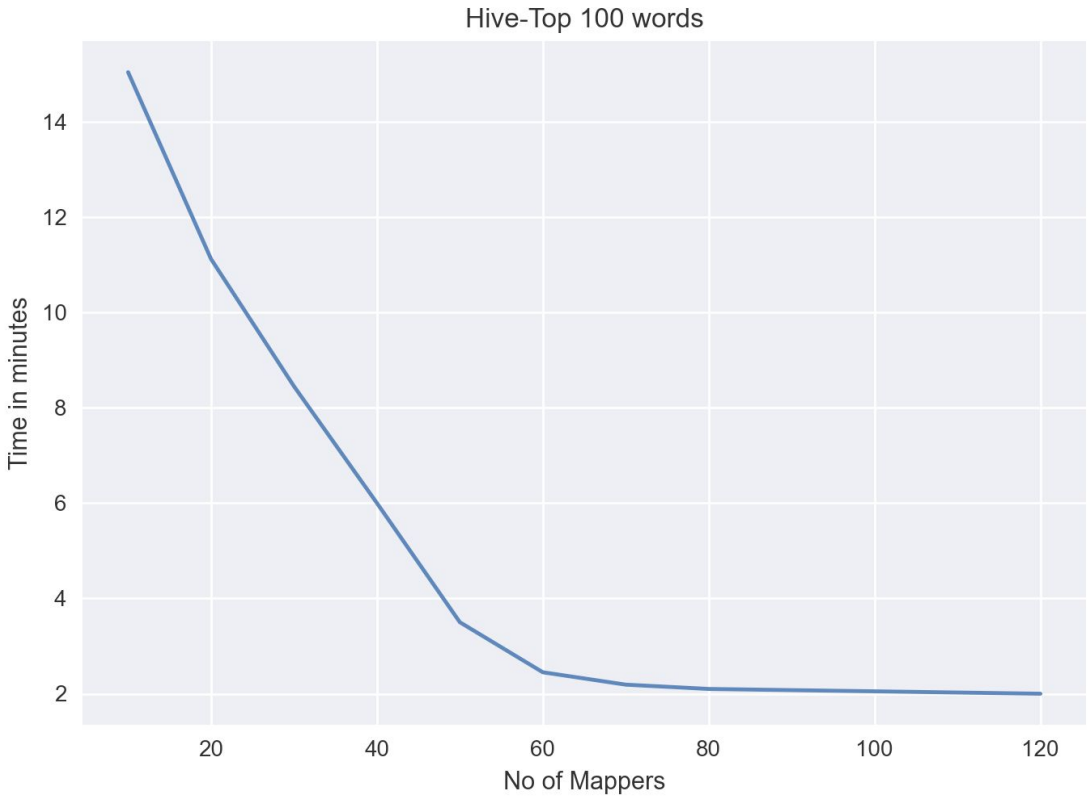
4. Number of Mappers vs Execution Time:

Number of mappers and reduces have been fixed and the execution time was poor when the number of mappers have been set to low. The split size was initiated from the maximum size of 32gb which used only 10 mappers. Gradually by decreasing the size of the split each time the mappers were assigned. We can see that from this table. The command that was used to split is set ***mapreduce.input.fileinputformat.split.maxsize***

These values have been plotted into a graph:

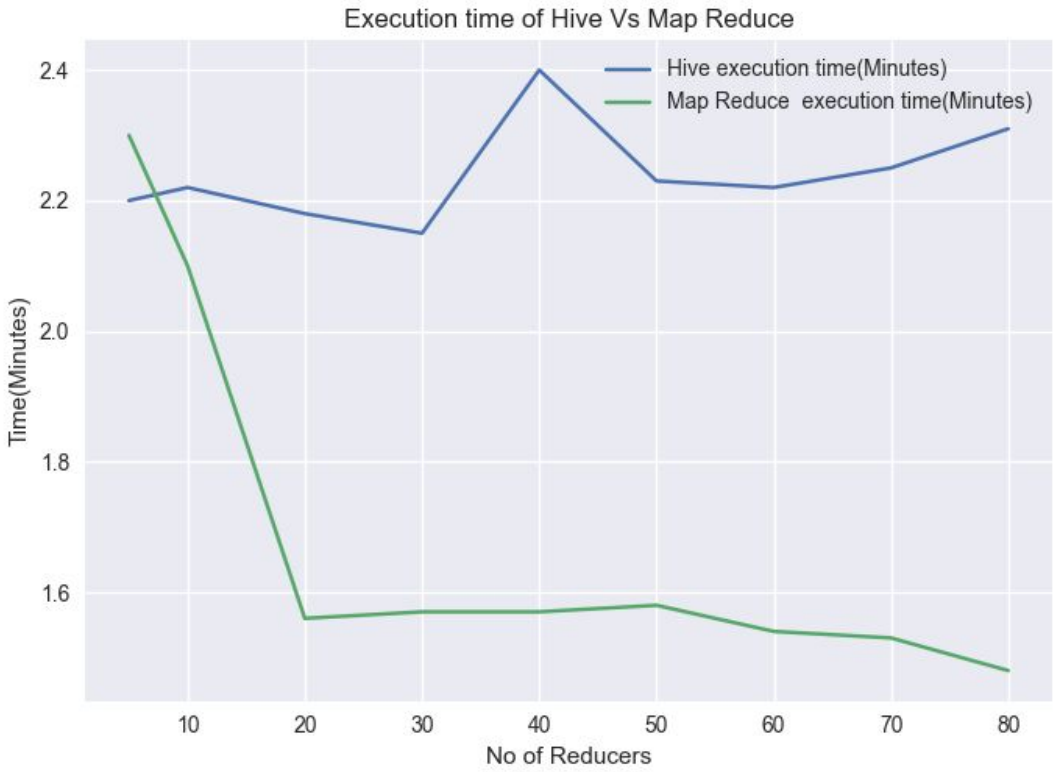
Number of mappers	Number of reducers	Execution time in Minutes
10	10	15.04
20	20	11.12
30	30	8.45
40	40	6.00
50	50	3.50

60	60	2.45
70	70	2.19
81	80	2.10
100	120	2.00
200	200	2.05



Comparison of Hive and MapReduce

After having worked on both Hive and Map Reduce, we came to the conclusion that Hive is much easier to code as compared to Map Reduce as Hive code mostly resembles the sql structure. Also Hive codes are short and thus easier to debug, maintain and make changes. The following graph has been plotted using the execution time we achieved for both Hive and Map Reduce for the number of reduces we used.


Execution time of Hive Vs Map Reduce
Hive execution time(Minutes)
Map Reduce execution time(Minutes)
Time(Minutes)
2.4
2.2
2.0
1.8
1.6
10
20
30
40
50
60
70
80
No of Reducers