

CS562 Assignment 1

Alexandros Kornilakis

October 2018

1 Introduction

At this assignment we are called to implement a number of simple tasks (find stopwords, create an inverted index) on Apache Hadoop Platform. To implement these tasks, I used the standard Java bindings in a testing environment. This environment consisted of a vagrant VM with CDH and a host machine with AMD FX-6300 and 8GB RAM.

2 Exercise 1 : Frequent Terms and Stop Words

To run the code for this exercise, go to exercisel directory and run make. You probably have to modify the username variable in the makefile to match the right one (cloudera).

The main method is in MRManager.java file where the job configurations hold. The mapper and reducer for the wordcount are at CountMap and CountReduce classes respectively, while the corresponding classes for Stopwords are StopwordsMap and StopwordsReduce. In order to sort the output by value, I created a custom Writable class for words as well as custom Partitioner and GroupingComparator.

The sorted stopwords list is at the file stopwords.csv. However, the 10 most frequent words sorted by their frequency (desc) are:

the	157207
and	124391
a	83592
of	80834
to	72891
i	58296
it	51992
in	48977
that	41497
was	39729

```

18/10/18 06:59:41 INFO mapreduce.Job: Counters: 50
File System Counters
  FILE: Number of bytes read=924937
  FILE: Number of bytes written=3081257
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=666867
  HDFS: Number of bytes written=549
  HDFS: Number of read operations=33
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=69287
Job Counters
  Launched map tasks=10
  Launched reduce tasks=1
  Data-local map tasks=10
  Total time spent by all maps in occupied slots (ms)=197239
  Total time spent by all reduces in occupied slots (ms)=660683
  Total time spent by all map tasks (ms)=197239
  Total time spent by all reduce tasks (ms)=660683
  Total vcore-seconds taken by all map tasks=197239
  Total vcore-seconds taken by all reduce tasks=660683
  Total megabyte-seconds taken by all map tasks=201972736
  Total megabyte-seconds taken by all reduce tasks=676539392
Map-Reduce Framework
  Map input records=69285
  Map output records=69285
  Map output bytes=786361
  Map output materialized bytes=924991
  Input split bytes=1390
  Combine input records=0
  Combine output records=0
  Reduce input groups=69285
  Reduce shuffle bytes=924991
  Reduce input records=69285
  Reduce output records=10
  Spilled Records=138570
  Shuffled Maps =10
  Failed Shuffles=0
  Merged Map outputs=10
  GC time elapsed (ms)=23687
  CPU time spent (ms)=475280
  Physical memory (bytes) snapshot=2483602432
  Virtual memory (bytes) snapshot=16518885376
  Total committed heap usage (bytes)=1716887552
RecordCounter
  Reducer-no-0=69285
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0

```

Figure 1: Screenshot of ex1 output

3 Exercise 2

3.1 a) Measuring the Performance on Map Reduce

1. 10 reducers and no Combiner. Execution time : 19470ms

```
File Edit View Search Terminal Help
18/10/19 07:00:57 INFO mapreduce.Job: map 77% reduce 0%
18/10/19 07:00:59 INFO mapreduce.Job: map 94% reduce 0%
18/10/19 07:01:01 INFO mapreduce.Job: map 100% reduce 0%
18/10/19 07:01:20 INFO mapreduce.Job: map 100% reduce 100%
18/10/19 07:01:21 INFO mapreduce.Job: Job job_153956036002_0000 completed successfully
18/10/19 07:01:22 INFO mapreduce.Job: Counters: 55

File System Counters
  FILE: Number of bytes read=3826276
  FILE: Number of bytes written=6754291
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=18414846
  HDFS: Number of bytes written=2527893
  HDFS: Number of read operations=21
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=2

Job Counters
  Launched map tasks=6
  Launched reduce tasks=1
  Data-local map tasks=6
  Total time spent by all maps in occupied slots (ms)=264333
  Total time spent by all reduces in occupied slots (ms)=24790
  Total time spent by all map tasks (ms)=264333
  Total time spent by all reduce tasks (ms)=24790
  Total vcore-seconds taken by all map tasks=264333
  Total vcore-seconds taken by all reduce tasks=24790
  Total megabyte-seconds taken by all map tasks=270676992
  Total megabyte-seconds taken by all reduce tasks=25384960

Map-Reduce Framework
  Map input records=325960
  Map output records=3200725
  Map output bytes=50723835
  Map output materialized bytes=2145562
  Input split bytes=822
  Combine input records=3200725
  Combine output records=78708
  Reduce input groups=78702
  Reduce shuffle bytes=2145562
  Reduce input records=78708
  Reduce output records=78702
  Spilled Records=216312
  Shuffled Maps=6
  Failed Shuffles=0
  Merged Map outputs=6
  GC time elapsed (ms)=4830
  CPU time spent (ms)=19470
  Physical memory (bytes) snapshot=1448333312
  Virtual memory (bytes) snapshot=10515668992
  Total committed heap usage (bytes)=1101619200

RecordCounter
  Reducer-no-0=181743
  Reducer-no-1=10362
  Reducer-no-2=5251
  Reducer-no-3=3892
  Reducer-no-4=965
  Reducer-no-5=142

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=18414824
File Output Format Counters
  Bytes Written=2527893
[cloudera@quickstart exercise2]$
```

2. 10 reducers and a Combiner. Execution time : 52580ms. The combiner reduces the number of data that are send to the reducer and thus provides a speedup. However, this may not always be the case. The use of combiner enforce an additional shuffle and sort phase and thus creates a trade-off. Furthermore, the use of 10 reducers for a single node machine imposes a significant latency due to the increased I/O requirements of the data trans-

```

Map output records=69285
Map output bytes=786361
Map output materialized bytes=1075
Input split bytes=1390
Combine input records=69285
Combine output records=100
Reduce input groups=100
Reduce shuffle bytes=1075
Reduce input records=100
Reduce output records=10
Spilled Records=200
Shuffled Maps =10
Failed Shuffles=0
Merged Map outputs=10
GC time elapsed (ms)=4900
CPU time spent (ms)=52580
Physical memory (bytes) snapshot=2212003840
Virtual memory (bytes) snapshot=16551325696
Total committed heap usage (bytes)=1716887552
RecordCounter
  Reducer-no-0=378
  Reducer-no-1=269
  Reducer-no-2=269
  Reducer-no-3=301
  Reducer-no-4=288
  Reducer-no-5=238
  Reducer-no-6=296
  Reducer-no-7=275
  Reducer-no-8=269
  Reducer-no-9=281
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=665477
File Output Format Counters
  Bytes Written=91
[cloudera@quickstart exercisel]$
fer.

```

3. 10 reducers, a Combiner and compressing the intermediate results. For compression I used the snappy codec. Execution time : 49590ms. I've noticed a decreased latency due to the compression. For larger file sizes, we could observe vastly improvements.

```

File Edit View Search Terminal Help
18/10/19 05:29:53 INFO mapreduce.Job: Counters: 68
File System Counters
  FILE: Number of bytes read=920
  FILE: Number of bytes written=1235852
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=166867
  HDFS: Number of bytes written=869
  HDFS: Number of read operations=33
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=3001
Job Counters
  Killed map tasks=1
  Launched map tasks=11
  Launched reduce tasks=1
  Data-local map tasks=11
  Total time spent by all maps in occupied slots (ms)=552604
  Total time spent by all reduces in occupied slots (ms)=56359
  Total time spent by all map tasks (ms)=552604
  Total time spent by all reduce tasks (ms)=56359
  Total vcore-seconds taken by all map tasks=552604
  Total vcore-seconds taken by all reduce tasks=56359
  Total megabyte-seconds taken by all map tasks=56386496
  Total megabyte-seconds taken by all reduce tasks=57711616
Map-Reduce Framework
  Map input records=69285
  Map output records=69285
  Map output bytes=786361
  Map output materialized bytes=1109
  Input split bytes=1390
  Combine input records=69285
  Combine output records=100
  Reduce input groups=100
  Reduce shuffle bytes=1109
  Reduce input records=100
  Reduce output records=10
  Spilled Records=200
  Shuffled Maps=10
  Failed Shuffles=0
  Merged Map outputs=10
  GC time elapsed (ms)=3867
  CPU time spent (ms)=48590
  Physical memory (bytes) snapshot=2338845184
  Virtual memory (bytes) snapshot=1661874272
  Total committed heap usage (bytes)=1716887352
RecordCounter
  Reducer-no-0=302
  Reducer-no-1=278
  Reducer-no-2=269
  Reducer-no-3=320
  Reducer-no-4=298
  Reducer-no-5=275
  Reducer-no-6=304
  Reducer-no-7=270
  Reducer-no-8=294
  Reducer-no-9=295
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=665477
File Output Format Counters
  Bytes Written=95
[cloudera@quickstart exercise]$

```

3.2 b) A variation of an inverted index.

1. At the following screenshot we can observe the first entries of the requested inverted index.

```
File Edit View Search Terminal Help
cloudera@quickstart exercise2$ hadoop fs -cat /user/cloudera/stopwords/intrm/* | more
1 1 a pg1100.txt pg1513.txt pg2253.txt pg100.txt pg1120.txt pg3200.txt
2 aa pg1513.txt pg100.txt
3 aar pg3200.txt
4 2 abate pg2253.txt
5 2 abide pg1120.txt
6 2 addition pg1100.txt
7 3 abn pg100.txt
8 3 aaron pg3200.txt
9 3 ab pg1513.txt
10 3 abbaisse pg2253.txt
11 3 abler pg1120.txt
12 3 after pg1100.txt
13 4 aac pg100.txt
14 4 aart pg3200.txt
15 4 abbreviations pg2253.txt
16 4 about pg1120.txt
17 4 ac pg1513.txt
18 4 all pg1100.txt
19 44146 a pg3200.txt
20 44147 aachen pg3200.txt
21 44148 abandon pg3200.txt
22 44149 abandoned pg3200.txt
23 44150 abate pg3200.txt
24 44151 abatement pg3200.txt
25 44152 abbey pg3200.txt
26 44153 abbot pg3200.txt
27 44154 abott pg3200.txt
28 44155 abbreviate pg3200.txt
29 44156 abbreviated pg3200.txt
30 44157 abe pg3200.txt
31 44158 abed pg3200.txt
32 44159 abet pg3200.txt
33 44160 abetted pg3200.txt
34 44161 abeyance pg3200.txt
35 44162 abhor pg3200.txt
36 44163 abide pg3200.txt
37 44164 abiding pg3200.txt
38 44165 abilities pg3200.txt
39 44166 ability pg3200.txt
40 44167 abject pg3200.txt
41 44168 ablaze pg3200.txt
42 44169 able pg3200.txt
43 44170 abler pg3200.txt
44 44171 ably pg3200.txt
45 44172 abner pg3200.txt
46 44173 abnormally pg3200.txt
47 44174 aboard pg3200.txt
48 44175 abolish pg3200.txt
49 44176 abolished pg3200.txt
50 44177 abolishing pg3200.txt
51 44178 abominable pg3200.txt
52 44179 abounded pg3200.txt
53 44180 abounding pg3200.txt
54 44181 about pg3200.txt
55 44182 above pg3200.txt
56 44183 abovementioned pg3200.txt
57 44184 abreast pg3200.txt
58 44185 abris pg3200.txt
59 44186 abroad pg3200.txt
60 44187 abrupt pg3200.txt
61 44188 abruptly pg3200.txt
62 44189 abscess pg3200.txt
63 44190 absence pg3200.txt
64 44191 absent pg3200.txt
65 44192 absentmindedness pg3200.txt
66 44193 absolute pg3200.txt
67 44194 absolutely pg3200.txt
68 44195 absorb pg3200.txt
```

2. The counter that reveals the number of unique words is named "reduce output records". I defined my own counter and the value of that counter was 78702. As a result, this is the number of unique words. T

4 Exercise 3. An extension of an inverted index

At this exercise we had to extend the inverted index in order to keep the frequency of each word in each document. It is obvious that that index is similar to the previous one. An example entry in our output set: 39 ability pg3200.txt144 pg1120.txt2 pg2253.txt3