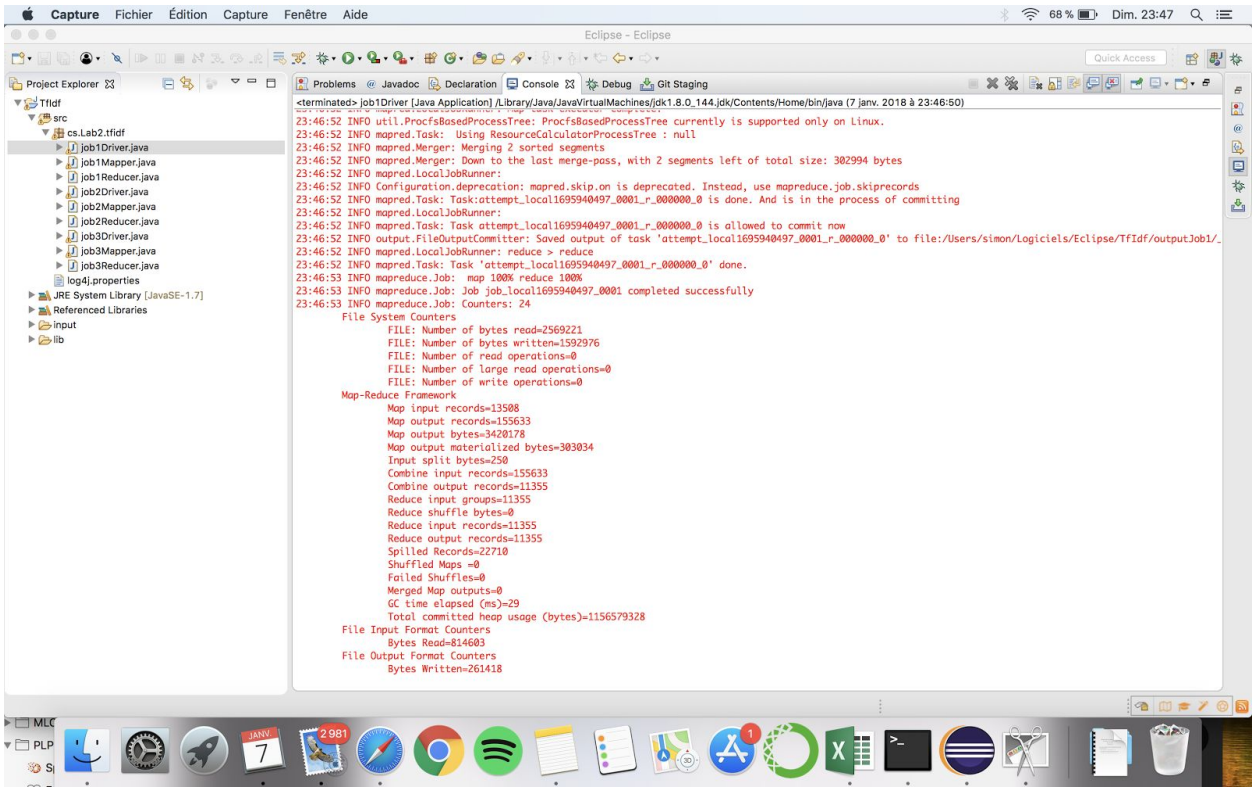
Rapport PLP

I. Hadoop

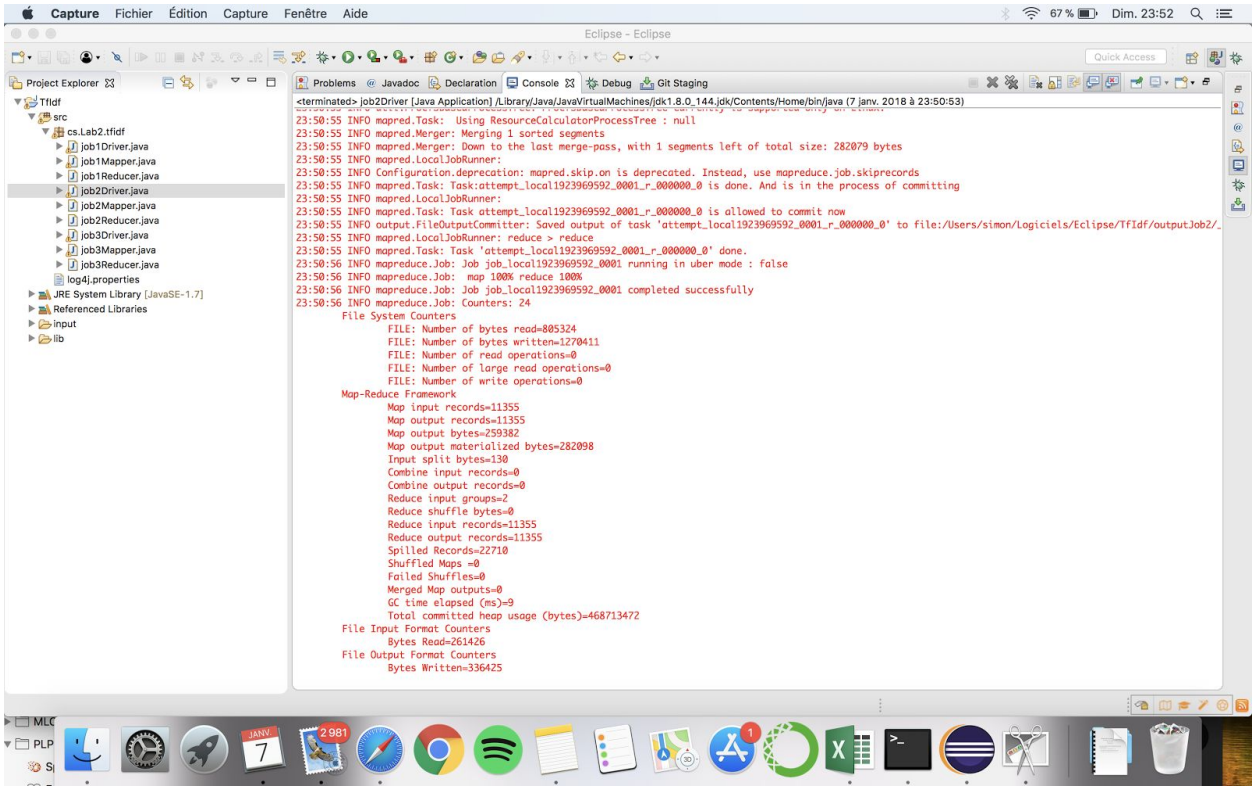
Q5.1

We did three MapReduce jobs to complete answer to the problem. Here is the screenshots of each job, launched one by one :

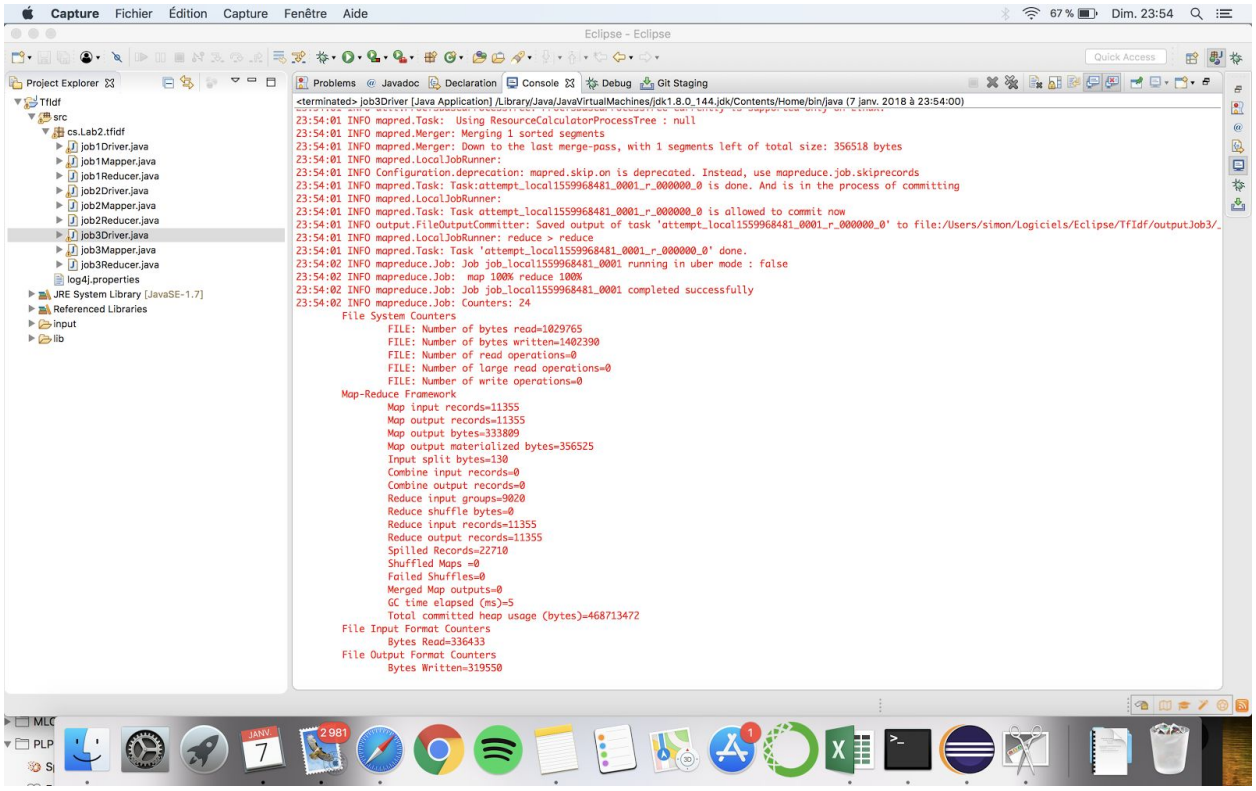
JOB1



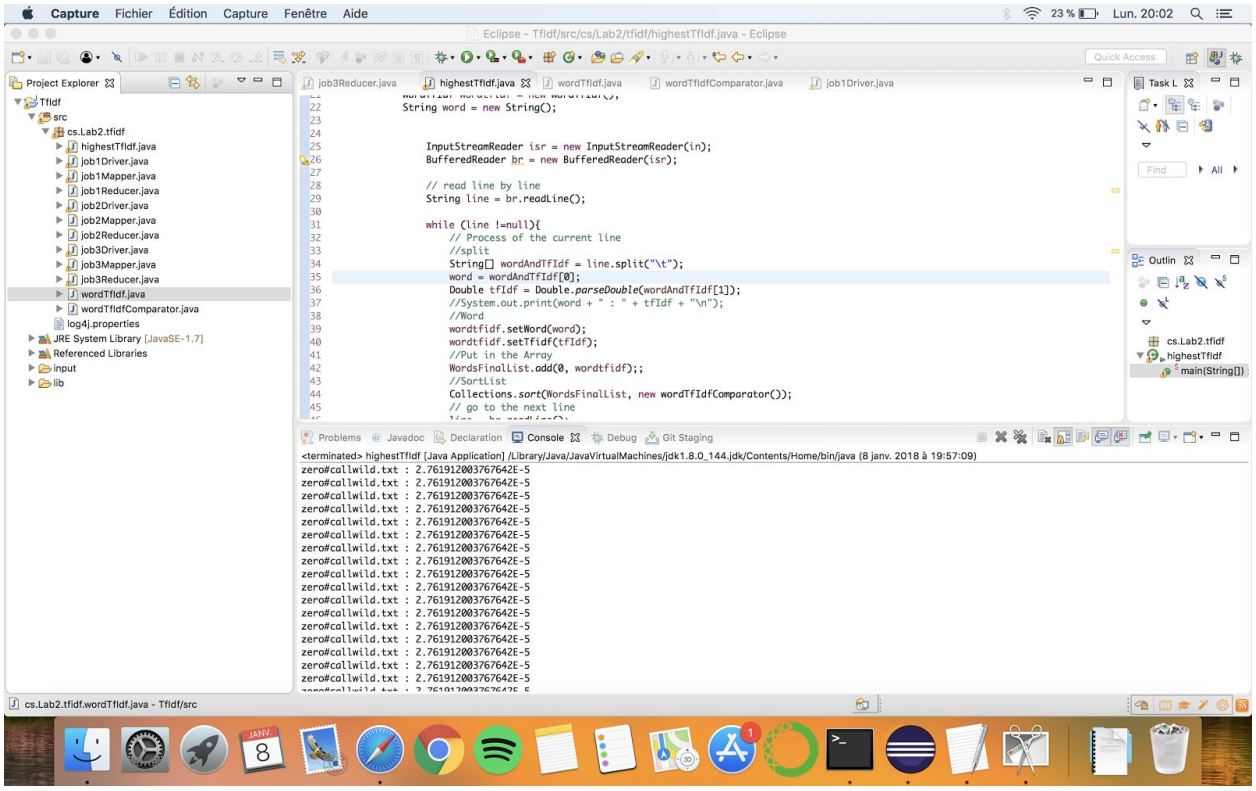
JOB2


Capture Fichier Édition Capture Fenêtre Aide
Dim. 23:52
Eclipse - Eclipse
Project Explorer
Problems Javadoc Declaration Console Debug Git Staging
<terminated> job2Driver [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (7 janv. 2018 à 23:50:53)
23:50:55 INFO mapred.Task: Task:attempt_local1923969592_0001_r_000000_0 is done. And is in the process of committing
23:50:55 INFO mapred.Task: Task attempt_local1923969592_0001_r_000000_0 is allowed to commit now
23:50:55 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1923969592_0001_r_000000_0' to file:/Users/simon/Logiciels/Eclipse/TfIdf/outputJob2/_
23:50:55 INFO mapred.Task: Task 'attempt_local1923969592_0001_r_000000_0' done.
23:50:56 INFO mapreduce.Job: Job job_local1923969592_0001 running in uber mode : false
23:50:56 INFO mapreduce.Job: map 100% reduce 100%
23:50:56 INFO mapreduce.Job: Job job_local1923969592_0001 completed successfully
23:50:56 INFO mapreduce.Job: Counters: 24
File System Counters
FILE: Number of bytes read=805324
FILE: Number of bytes written=1270411
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=11355
Map output records=11355
Map output bytes=259382
Map output materialized bytes=282098
Input split bytes=130
Combine input records=0
Combine output records=0
Reduce input groups=2
Reduce shuffle bytes=0
Reduce input records=11355
Reduce output records=11355
Spilled Records=22710
Shuffled Maps =0
Failed Shuffles=0
Merged Map outputs=0
GC time elapsed (ms)=9
Total committed heap usage (bytes)=468713472
File Input Format Counters
Bytes Read=261426
File Output Format Counters
Bytes Written=336425

JOB3

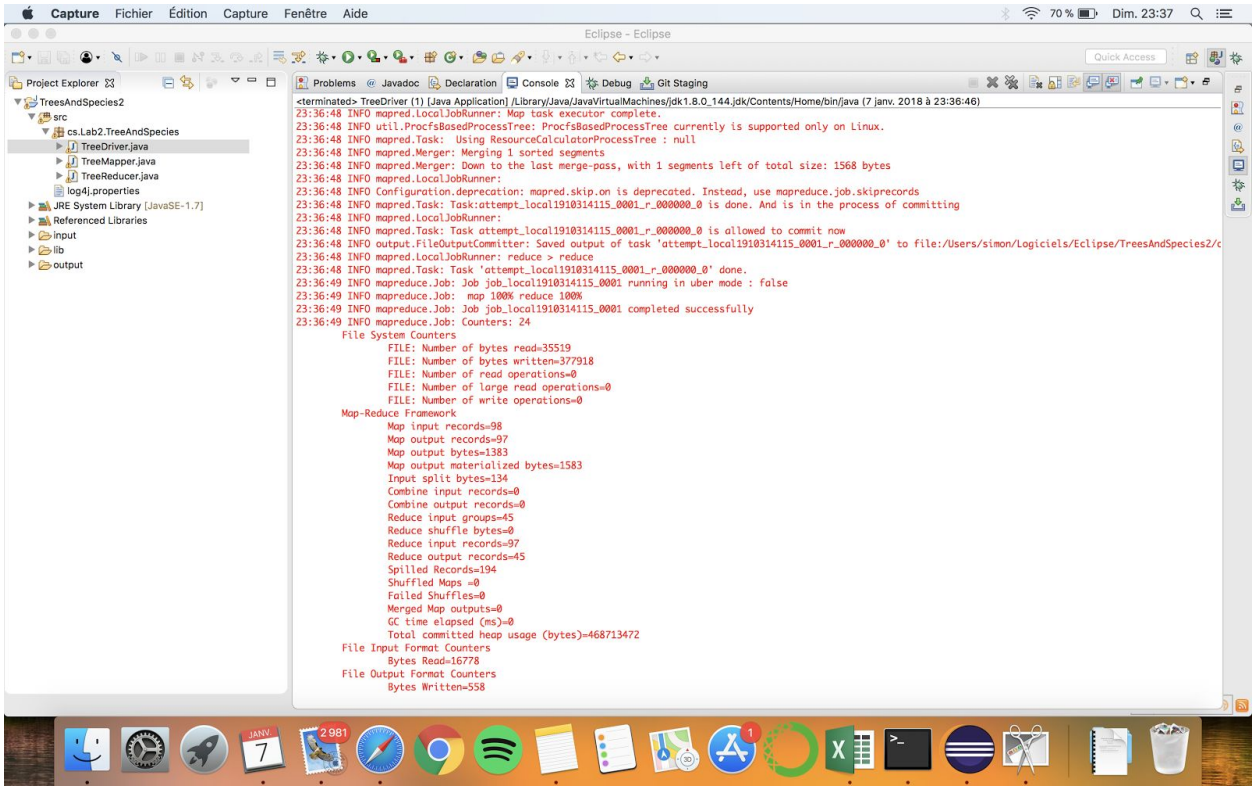

Capture Fichier Édition Capture Fenêtre Aide
Dim. 23:54
Eclipse - Eclipse
Project Explorer
Problems Javadoc Declaration Console Debug Git Staging
<terminated> job3Driver [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (7 janv. 2018 à 23:54:00)
23:54:01 INFO mapred.Task: Task:attempt_local1559968481_0001_r_000000_0 is done. And is in the process of committing
23:54:01 INFO mapred.Task: Task attempt_local1559968481_0001_r_000000_0 is allowed to commit now
23:54:01 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1559968481_0001_r_000000_0' to file:/Users/simon/Logiciels/Eclipse/TfIdf/outputJob3/_
23:54:01 INFO mapred.Task: Task 'attempt_local1559968481_0001_r_000000_0' done.
23:54:02 INFO mapreduce.Job: Job job_local1559968481_0001 running in uber mode : false
23:54:02 INFO mapreduce.Job: map 100% reduce 100%
23:54:02 INFO mapreduce.Job: Job job_local1559968481_0001 completed successfully
23:54:02 INFO mapreduce.Job: Counters: 24
File System Counters
FILE: Number of bytes read=1029765
FILE: Number of bytes written=1402390
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=11355
Map output records=11355
Map output bytes=333809
Map output materialized bytes=356525
Input split bytes=130
Combine input records=0
Combine output records=0
Reduce input groups=9020
Reduce shuffle bytes=0
Reduce input records=11355
Reduce output records=11355
Spilled Records=22710
Shuffled Maps =0
Failed Shuffles=0
Merged Map outputs=0
GC time elapsed (ms)=5
Total committed heap usage (bytes)=468713472
File Input Format Counters
Bytes Read=336433
File Output Format Counters
Bytes Written=319550

In the end I don't know why but I couldn't compute the ordered list of TFIDF. I don't know why but when adding one element to my arrayList, the array List become a List of that unique element, see the screenshot :

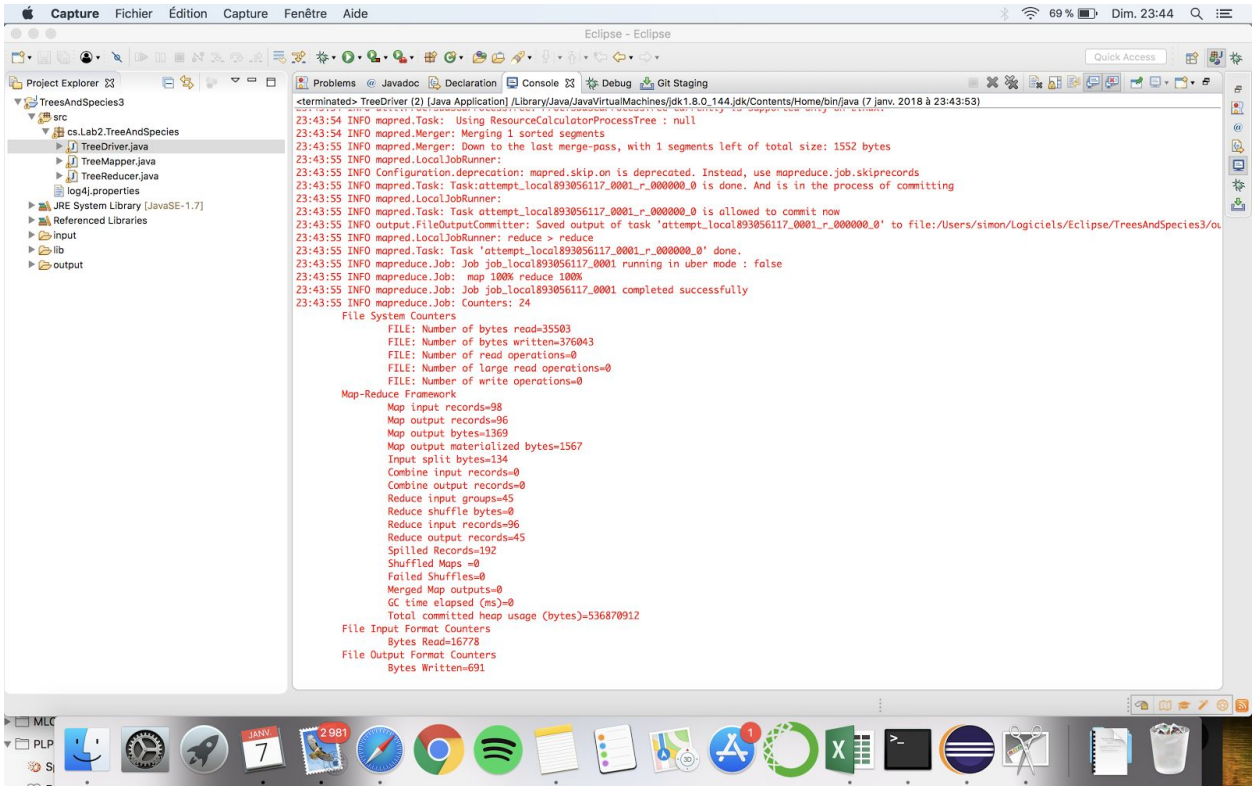


5.3 :

First Part :


Capture Fichier Édition Capture Fenêtre Aide
70 % Dim. 23:37
Eclipse - Eclipse
Quick Access
Project Explorer
TreesAndSpecies2
src
cs.Lab2.TreeAndSpecies
TreeDriver.java
TreeMapper.java
TreeReducer.java
log4j.properties
JRE System Library [JavaSE-1.7]
Referenced Libraries
input
lib
output
Problems Javadoc Declaration Console Debug Git Staging
<terminated> TreeDriver (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (7 janv. 2018 à 23:36:46)
23:36:48 INFO mapred.LocalJobRunner: Map task executor complete.
23:36:48 INFO util.ProcfsBasedProcessTree: ProcfsBasedProcessTree currently is supported only on Linux.
23:36:48 INFO mapred.Task: Using ResourceCalculatorProcessTree : null
23:36:48 INFO mapred.Merger: Merging 1 sorted segments
23:36:48 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 1568 bytes
23:36:48 INFO mapred.LocalJobRunner:
23:36:48 INFO Configuration.deprecation: mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
23:36:48 INFO mapred.Task: Task:attempt_local1910314115_0001_r_000000_0 is done. And is in the process of committing
23:36:48 INFO mapred.LocalJobRunner:
23:36:48 INFO mapred.Task: Task attempt_local1910314115_0001_r_000000_0 is allowed to commit now
23:36:48 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1910314115_0001_r_000000_0' to file:/Users/simon/Logiciels/Eclipse/TreesAndSpecies2/c
23:36:48 INFO mapred.LocalJobRunner: reduce > reduce
23:36:48 INFO mapred.Task: Task 'attempt_local1910314115_0001_r_000000_0' done.
23:36:49 INFO mapreduce.Job: Job job_local1910314115_0001 running in uber mode : false
23:36:49 INFO mapreduce.Job: map 100% reduce 100%
23:36:49 INFO mapreduce.Job: Job job_local1910314115_0001 completed successfully
23:36:49 INFO mapreduce.Job: Counters: 24
File System Counters
FILE: Number of bytes read=35519
FILE: Number of bytes written=377918
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=98
Map output records=97
Map output bytes=1383
Map output materialized bytes=1583
Input split bytes=134
Combine input records=0
Combine output records=0
Reduce input groups=45
Reduce shuffle bytes=0
Reduce input records=97
Reduce output records=45
Spilled Records=194
Shuffled Maps =0
Failed Shuffles=0
Merged Map outputs=0
GC time elapsed (ms)=0
Total committed heap usage (bytes)=468713472
File Input Format Counters
Bytes Read=16778
File Output Format Counters
Bytes Written=558

Second Part :


Capture Fichier Édition Capture Fenêtre Aide
Dim. 23:44
Eclipse - Eclipse
Project Explorer
TreesAndSpecies3
src
cs.Lab2.TreeAndSpecies
TreeDriver.java
TreeMapper.java
TreeReducer.java
log4j.properties
JRE System Library [JavaSE-1.7]
Referenced Libraries
input
lib
output
Problems Javadoc Declaration Console Debug Git Staging
<terminated> TreeDriver (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (7 janv. 2018 à 23:43:53)
23:43:54 INFO mapred.Task: Using ResourceCalculatorProcessTree : null
23:43:54 INFO mapred.Merger: Merging 1 sorted segments
23:43:55 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 1552 bytes
23:43:55 INFO mapred.LocalJobRunner:
23:43:55 INFO Configuration.deprecation: mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
23:43:55 INFO mapred.Task: Task:attempt_local893056117_0001_r_000000_0 is done. And is in the process of committing
23:43:55 INFO mapred.LocalJobRunner:
23:43:55 INFO mapred.Task: Task attempt_local893056117_0001_r_000000_0 is allowed to commit now
23:43:55 INFO output.FileOutputCommitter: Saved output of task 'attempt_local893056117_0001_r_000000_0' to file:/Users/simon/Logiciels/Eclipse/TreesAndSpecies3/ou
23:43:55 INFO mapred.LocalJobRunner: reduce > reduce
23:43:55 INFO mapred.Task: Task 'attempt_local893056117_0001_r_000000_0' done.
23:43:55 INFO mapreduce.Job: Job job_local893056117_0001 running in uber mode : false
23:43:55 INFO mapreduce.Job: map 100% reduce 100%
23:43:55 INFO mapreduce.Job: Job job_local893056117_0001 completed successfully
23:43:55 INFO mapreduce.Job: Counters: 24
File System Counters
FILE: Number of bytes read=35503
FILE: Number of bytes written=376043
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=98
Map output records=96
Map output bytes=1369
Map output materialized bytes=1567
Input split bytes=134
Combine input records=0
Combine output records=0
Reduce input groups=45
Reduce shuffle bytes=0
Reduce input records=96
Reduce output records=45
Spilled Records=192
Shuffled Maps =0
Failed Shuffles=0
Merged Map outputs=0
GC time elapsed (ms)=0
Total committed heap usage (bytes)=536870912
File Input Format Counters
Bytes Read=16778
File Output Format Counters
Bytes Written=691

II. Spark

A. Part 1 :Starter

1. *Print the top 10 most frequent words with their probability of appearance*

Script : *wordCount.py*

We use the .split() methods to split the lines into words. However that does not take into account special characters. It means : “the.” or “the:” is not taken into account to count the appearance of the.

2. *Get rid of special characters (.,:!?’)*

Script : *wordCountSpeChar.py*

To get rid of the special characters, we call the replace method as follow :

```
line.replace('.',' ')\
        .replace(',',' ')\
        .replace(':',' ')\
        .replace('!',' ')\
        .replace('?',' ')\
        .replace('\"',' ')\
        .split()
```

3. *Identify the transformations and the actions in our script*

Transformations : flatMap : to split lines into words and associate each word to a key value.

Action : reduceByKey : gather the key together and sum the key values.

Count: return the number of element in the dataset

4. *How many times are the transformations evaluated*

The transformations are evaluated for each action. In this case twice.

5. *Can you reduce this number ?*

We can persist the datasets to cache the results of transformations and reuse them in other actions on these datasets.

B. Part 1 :Intermediate

1. *Print the top 10 words from the Iliad that have “most disappeared” in The Odyssey*
2. *Do the same by swapping the Illiad and The Odyssey*
3. *Improve your script by getting rid of stopwords*

Script : *wordDisappeared.py*

4. *Use the Spark UI to make your script faster*

C. Part 2 : warmup

1. Create a cluster of 4-6 Spark nodes

Command on windows :

- Masternode : bin\spark-class org.apache.spark.deploy.master.Master
- Slavenodes : spark-class org.apache.spark.deploy.worker.Worker spark://ip:port

2. If possible, launch a couple HDFS datanodes
3. Launch wordcount.py on iliad.mb.txt

bin\spark-submit --master spark://192.168.0.17:7077 wordCount.py iliad100.txt

4. Launch wordcount.py ib raw.txt
5. Launch two jobs at the same time. Make them run at the same time

The master process only one at a time, and the other one is “waiting”.

6. What happens when a Spark node is brutally shutdown

The master notice the a slavenote shut down : “Lost executor 0 on 192.168.0.17: worker lost”
Then the master redistribute the work and still manage to complete the job thanks to the other workers.

D. Part 2 : Intermediate

1. What is Word2Vec ?

It is a technique that convert words into numeric vectors which can then be “fed into” various machine learning models to perform natural language processing.

2. Create a Word2Vec model of the Iliad
3. Who is Achilles + (Priam - Hector)

Script : *word2Vec.py*

E. Part 3 : Velib

1. *Create a Spark Streaming app that reads data from velib.behmo.com*

Any station has its “contract_name” + “name” that identify the station.
We have several times the same station for each RDDs. We use the “.transform(lambda rdd : rdd.distinct())” to solve this issue.

2. *Every 5 sec : print the empty Velib stations*

Script: *velibEmpty.py*

3. Every 5s: print the Velib stations that have become empty

Script: *velibGetEmpty.py*
We use the“.updateStateByKey” method that stores the following information:

-1 : Station just got empty
0 : Station is empty but haven’t just turn empty during the last batch
1 : Station is not empty

Then we can filter by value to get the stations that just got empty.

4. *Every 1 min: print the stations that were most active during the last 5 min (activity = number of bikes borrowed and return)*

Script: velibMostActive.py

We use “.reduceByKeyAndWindow” method to take into account a 5 min window and to print the results every minute.