Homework #2 - Spark

Part 1 - Implement and Analyze Word Count

Analysis

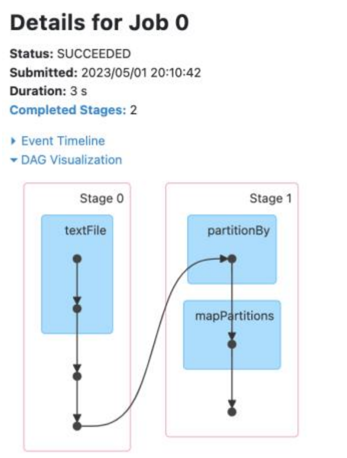
1. In the PySpark REPL, run your basic word count program on a single text file.

a. What are the 25 most common words? Include a screenshot of program output to back-up your claim.

The 25 most common words for one textbook are:

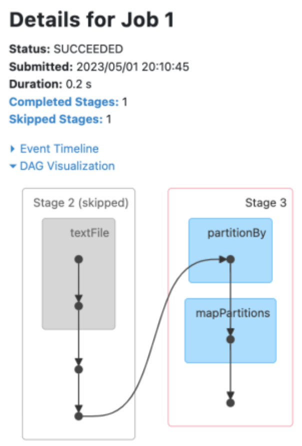
```
('bunny', 678)
('patter', 434)
('sue', 427)
('dog', 328)
('brown', 279)
('mr', 229)
('asked', 163)
('trick', 153)
('she', 151)
('one', 150)
('“i', 143)
('“oh', 129)
('get', 127)
('back', 125)
('cried', 119)
('george', 118)
('little', 117)
('boys', 117)
('mrs', 106)
('go', 98)
('going', 95)
('toby', 94)
('children', 92)
('make', 91)
('went', 91)
```

b. How many stages is execution broken up into? Explain why. Include a screenshot of the DAG visualization from Spark's WebUI to back-up your claim.



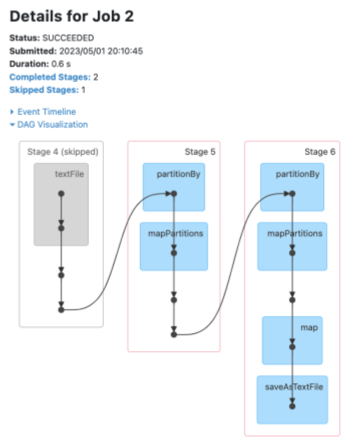
Details for Job 0

Status: SUCCEEDED
Submitted: 2023/05/01 20:10:42
Duration: 3 s
Completed Stages: 2



Details for Job 1

Status: SUCCEEDED
Submitted: 2023/05/01 20:10:45
Duration: 0.2 s
Completed Stages: 1
Skipped Stages: 1



Details for Job 2

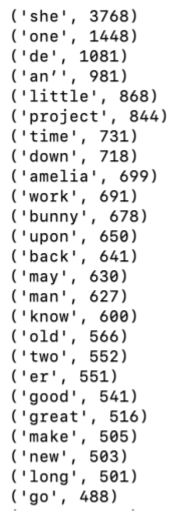
Status: SUCCEEDED
Submitted: 2023/05/01 20:10:45
Duration: 0.6 s
Completed Stages: 2
Skipped Stages: 1

There are three jobs altogether in the DAG. The total execution is divided into three stages, The first stage is reading the file, where the data from source file is read into a RDD, In the second stage the data is counted by words and then sorted based on the number of occurrences. In the last stage the data is written into an output directory using saveTextFile after reducing.

2. In the PySpark REPL, run your extended word count program on all 10 text files.

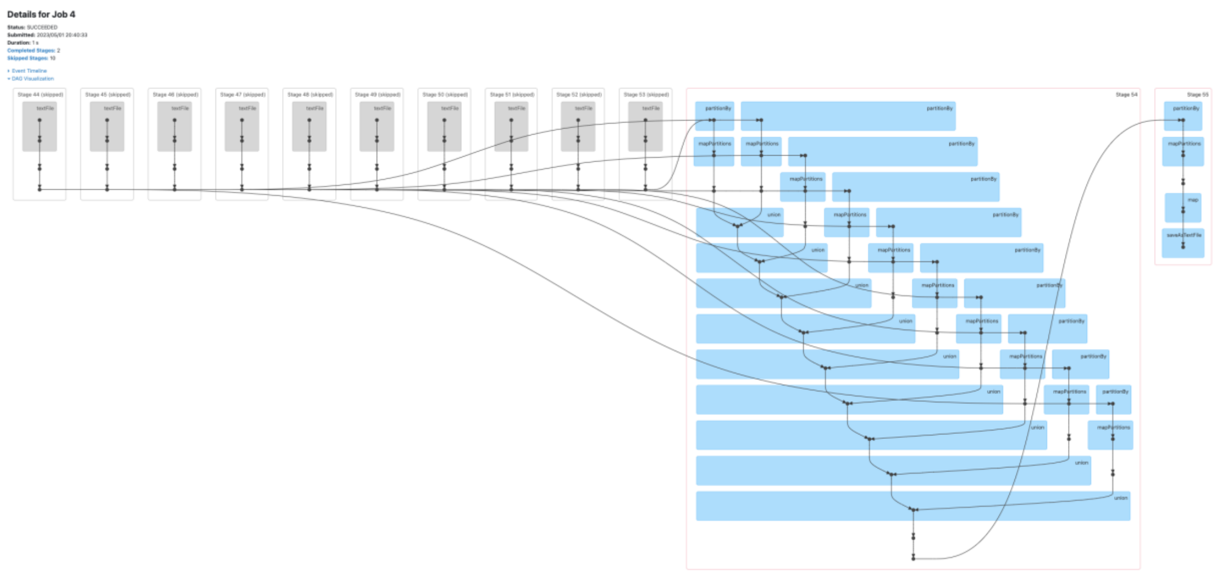
a. What are the 25 most common words? Include a screenshot of program output to back-up your claim.

The 25 most common words for 10 textbooks are:



```
('she', 3768)
('one', 1448)
('de', 1081)
('an'', 981)
('little', 868)
('project', 844)
('time', 731)
('down', 718)
('amelia', 699)
('work', 691)
('bunny', 678)
('upon', 650)
('back', 641)
('may', 630)
('man', 627)
('know', 600)
('old', 566)
('two', 552)
('er', 551)
('good', 541)
('great', 516)
('make', 505)
('new', 503)
('long', 501)
('go', 488)
```

b. How many stages is execution broken up into? Explain why. Include a screenshot of the DAG visualization from Spark's WebUI to back-up your claim.



Details for Job 4

Status: SUCCEEDED
Submitted: 2023/05/01 20:40:33
Duration: 1 s
Completed Stages: 2
Skipped Stages: 10

▸ Event Timeline
▾ DAG Visualization

There are five jobs altogether in the DAG. The total execution is divided into twelve stages. Each file is executed independently, and the transformations are performed lazily until an action is triggered. The first 10 stages are reading the source files, where the data from source file is read into a RDD, In the 11th stage the data is counted by words and then sorted based on the number of occurrences. In the last stage the data is written into an output directory using saveTextFile after reducing.

3. Answer the following based on your knowledge of both MapReduce and Spark:
a. If you were running your WordCount programs in a large cluster or cloud environment, and one of the nodes you were running on died mid computation, how would your MapReduce and Spark programs handle this?

Both MapReduce and Spark are built to manage node failures in a distributed computing environment, such as a big cluster or cloud environment.

When a node fails in MapReduce, the framework automatically recognizes the failure and re-runs the unsuccessful jobs on another node. To reduce how much a node failure affects the program's overall performance, the framework keeps track of job progress and data locality information.

When a node in Spark fails, the Spark framework recognizes the failure and computes the lost RDD partitions again on a different node. With the aid of RDD lineage information and fault-tolerant storage for intermediate data, Spark can recover from node failures and carry on with computations without having to completely restart the application.

Both implement fault tolerance through the replication of intermediate data and results across multiple nodes, ensuring that even in the event of a node failure, the computation can still be completed using the replicated data on other nodes.

b. Explain one concrete benefit you experienced when writing the Spark version of WordCount compared to the MapReduce version.

One concrete benefit when writing the Spark version of WordCount compared to the MapReduce version is the faster development and debugging time.

The code is easier to comprehend and alter because it is written in Spark in a more condensed and expressive manner. As an illustration, rather than needing to write numerous MapReduce jobs in different files, we may chain together various transformations and operations on an RDD in a single line of code. We can test our code more easily and fast as a result.

Additionally, Spark provides an interactive shell (called the Spark shell) that enables us to test and experiment with our code live. To quickly find mistakes and correct them, we can load the data, perform transformations, and analyze the outcomes in real-time.

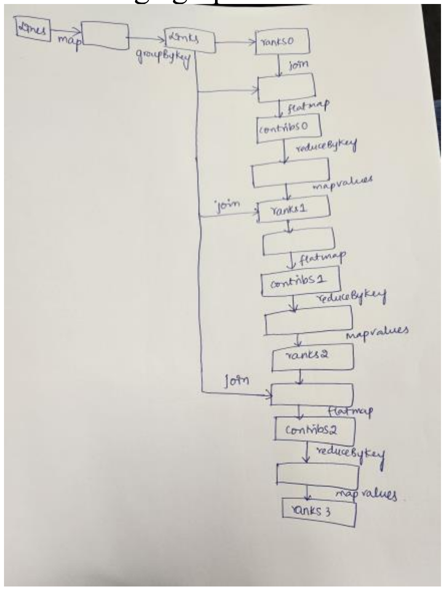
Contrarily, with MapReduce, we must construct numerous tasks and manually manage the intermediate output, which can be laborious and error prone. MapReduce jobs must also be built and deployed on a Hadoop cluster before they can be used, which slows down the development and debugging process and adds additional complexity.

Overall, compared to the MapReduce version, the quicker development and debugging times in Spark enable us to iterate more quickly and deliver solutions.

Part 2 - Code Analysis

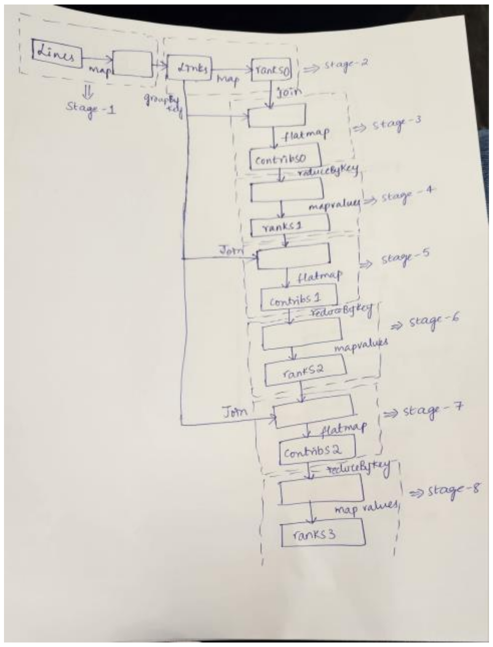
4. Given the above spark application, draw the lineage graph DAG for the RDD ranks on line 12 when the iteration variable i has a value of 2. Include nodes for all intermediate RDDs, even if they are unnamed.

The Lineage graph DAG for the RDD ranks on line 2 with i=2 is here:



5. How many stages will the above DAG be broken into? Give the number of stages AND draw stage boundaries on your diagram.

The DAG is further broken into stages, and it is given below. The number of stages is 8.



6. Identify in the above code (by function name AND line number) one instance of:

a. A transformation that results in a wide dependency
groupByKey() or reduceByKey() and line numbers are 3 and 12

b. A transformation that results in a narrow dependency
map(), flatMap(), mapValues() and line numbers are 2,6,10,13

c. A transformation that may result in a narrow dependency OR a wide dependency.
join() and the line number is 9

d. An action
count() and the line number is 5

7. How many "jobs" will the above code run if iters has value 10?
The number of jobs for the given code is one,if the iters has value 10.As we only have one action i.e., count()

8. What algorithm is the above code an implementation of?
PageRank Algorithm