

Big Data Frameworks CSE3120

Lab – 7 Spark Word Count Program

Name: Naveen Nidadavolu

Roll No: 22MIA1049

Aim

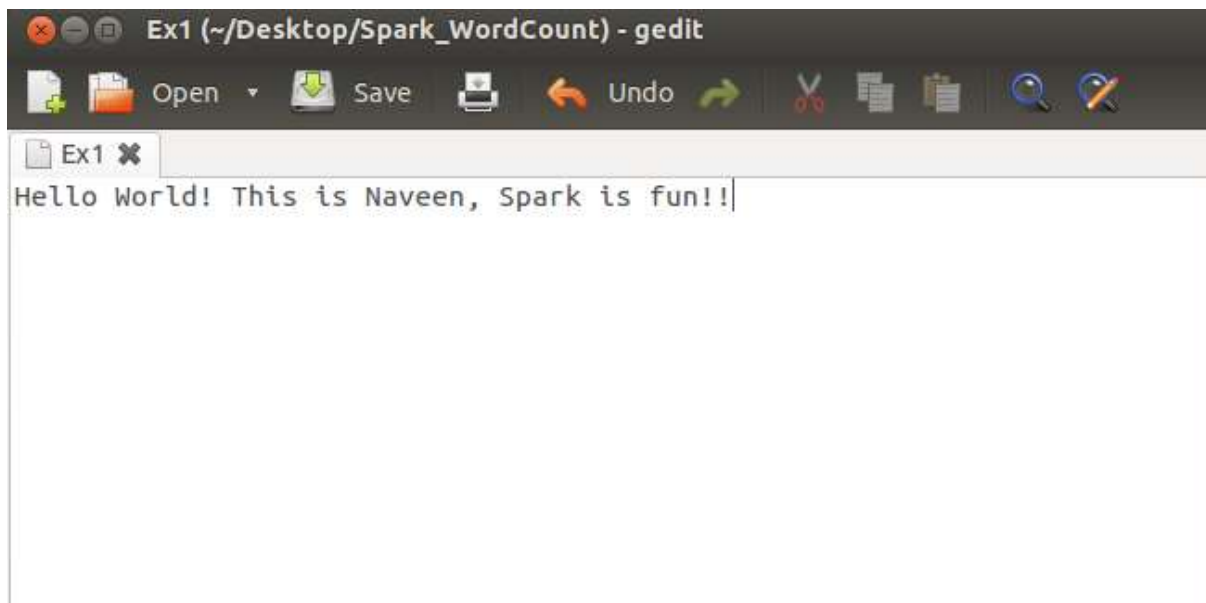
To implement a Word Count program using Apache Spark to count the occurrences of words in a given text file.

Algorithm

1. **Start**
2. Initialize the Spark environment.
3. Load the input text file into an RDD (Resilient Distributed Dataset).
4. Split each line of the text file into individual words.
5. Map each word to a key-value pair in the form of (word, 1).
6. Use the **reduceByKey()** function to sum up the counts of each word.
7. Collect the final word count results.
8. Display the results.
9. **End**

Procedure

1. Create a sample text file named "Ex1" and save it on the desktop.



2. . Give the command: spark-shell, to check the spark framework version

```
25/03/24 11:08:07 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
25/03/24 11:08:07 WARN Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interface eth0)
25/03/24 11:08:07 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
25/03/24 11:08:10 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1742794687968).
Spark session available as 'spark'.
Welcome to

      / _ \   / _ \   / _ \   / _ \
     / ___\ / ___\ / ___\ / ___\
    / ____ \| ____ \| ____ \| ____ \|
   / /    \| /    \| /    \| /    \|
  /_/      \/_/      \/_/      \/_/

version 2.2.0

Using Scala version 2.11.8 (Java HotSpot(TM) Client VM, Java 1.8.0_45)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Apache Spark is an open source, in-memory distributed computing engine created to address the problem of processing large datasets for data analytics and machine learning. Spark is written in Scala and it's native integration with Spark APIs

3. Execute Word Count in Spark

- Load the text file Ex1 into Spark
- Split the lines into words, map each word to a key-value pair, and count occurrences
- Display the word count results

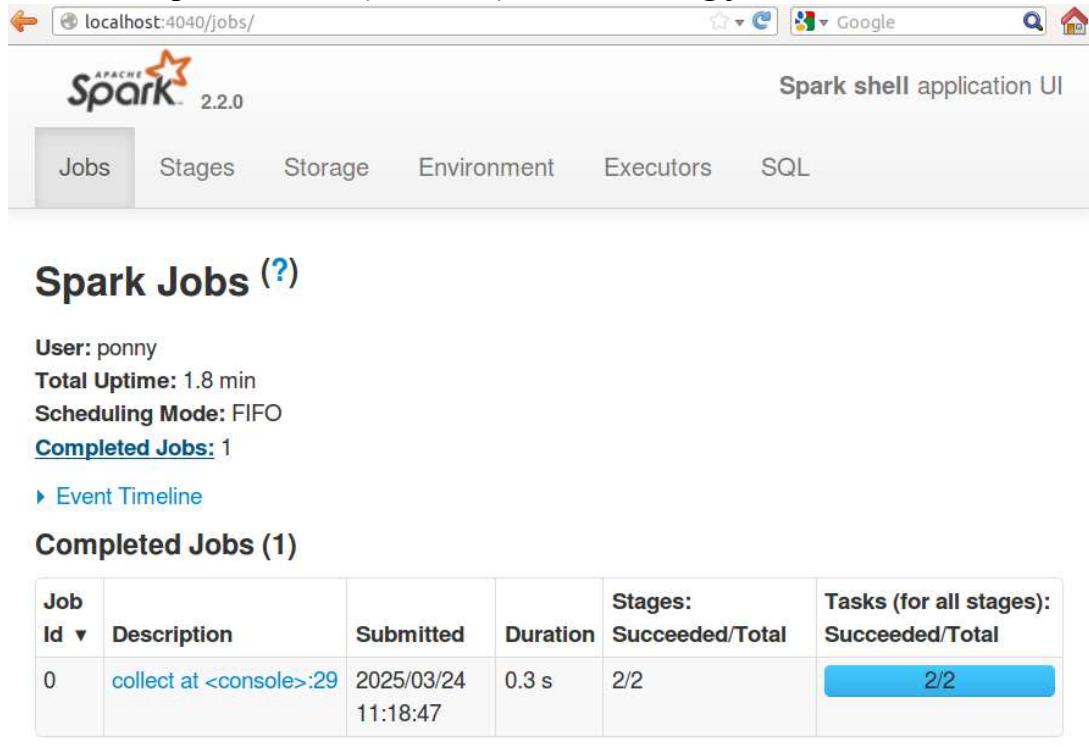
```
scala> var a = sc.textFile("/home/ponny/Desktop/Ex1").flatMap(line => line.split
(" ")).map(word => (word,1))
25/03/24 11:18:34 WARN SizeEstimator: Failed to check whether UseCompressedOops
is set; assuming yes
a: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at <cons
ole>:24

scala> var b = a.reduceByKey(_ + _);
b: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <c
onsole>:26

scala> b.collect
res0: Array[(String, Int)] = Array((fun!!,1), (Spark,1), (is,2), (Hello,1), (Nav
een,,1), (World!,1), (This,1))

scala>
```

4. Check the Spark Web UI (local host) for monitoring job execution and results.



The screenshot shows the Apache Spark Web UI at localhost:4040/jobs/. The interface includes a navigation bar with tabs for Jobs, Stages, Storage, Environment, Executors, and SQL. The 'Jobs' tab is active, displaying the 'Spark Jobs (?)' section. This section provides summary information: User: ponny, Total Uptime: 1.8 min, Scheduling Mode: FIFO, and Completed Jobs: 1. A link for 'Event Timeline' is also present. Below this, the 'Completed Jobs (1)' section contains a table with one job entry.

Job Id ▼	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	collect at <console>:29	2025/03/24 11:18:47	0.3 s	2/2	2/2

localhost:4040/jobs/job/?id=0

Spark 2.2.0 Spark shell application UI

Jobs Stages Storage Environment Executors SQL

Details for Job 0

Status: SUCCEEDED
Completed Stages: 2

▶ Event Timeline
▶ DAG Visualization

Completed Stages (2)

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
1	collect at <console>:29 +details	2025/03/24 11:18:48	30 ms	1/1			104.0 B	
0	map at <console>:24 +details	2025/03/24 11:18:47	0.2 s	1/1	44.0 B			104.0 B

Spark 2.2.0 Jobs Stages Storage Environment Spark shell application UI

Details for Job 0

Status: SUCCEEDED
Completed Stages: 2

▶ Event Timeline
▼ DAG Visualization

```

graph TD
    A[Stage 0] --> B[textFile]
    B --> C[flatMap]
    C --> D[map]
  
```

The DAG visualization for Stage 0 shows a linear sequence of three operations: 'textFile', 'flatMap', and 'map'. Each operation is represented by a blue rectangular box, and they are connected by downward-pointing arrows, indicating the flow of data from the input file through the transformations to the final output.

Results:

The Word Count program in Spark successfully counts the occurrences of words in the given input file.