ASSIGNMENT – 04

Introduction to Distributed Systems IS41243

Spark shell word count

By:
Sayanthiny.R
15APC2383

DEPARTMENT OF COMPUTING & INFORMATION SYSTEMS
FACULTY OF APPLIED SCIENCES
SABARAGAMUWA UNIVERSITY OF SRI LANKA

TABLE OF CONTENTS

Table of figures

Spark shell

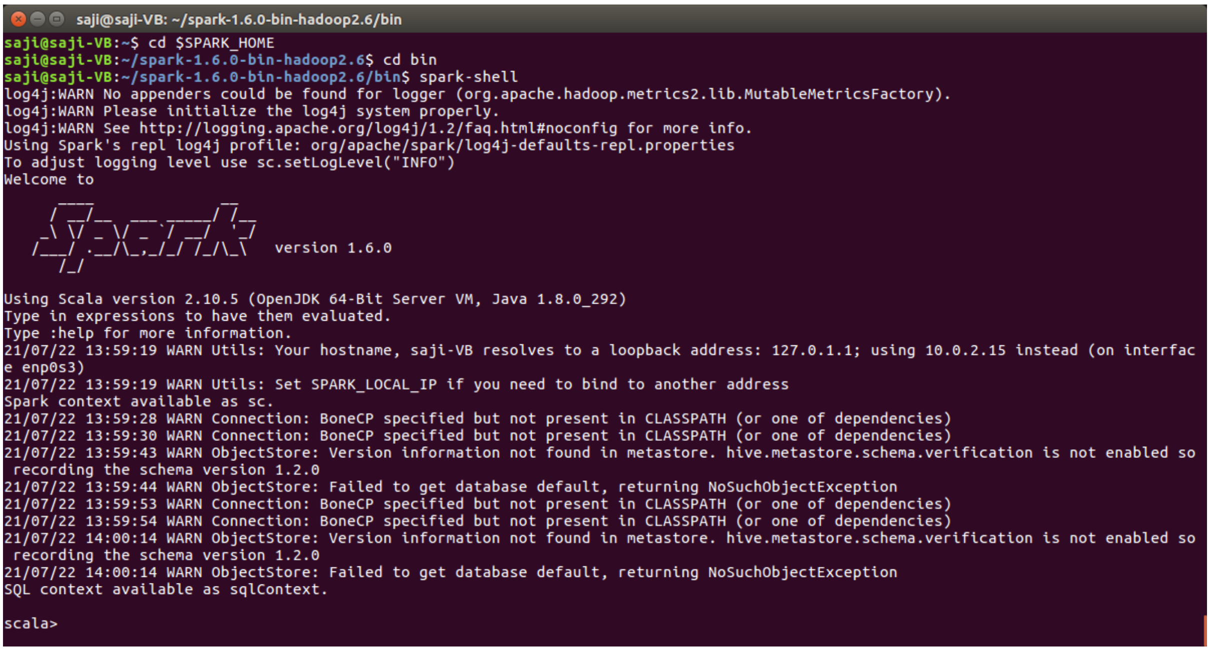
The spark shell is launched by the command spark-shell..

- cd $SPARK_HOME
- cd bin

Step 1: launch the Scala Spark shell,

To launch the Scala enter the following command.

```
$ spark-shell.
```



```
saji@saji-VB:~$ cd $SPARK_HOME
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6$ cd bin
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin$ spark-shell
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's repl log4j profile: org/apache/spark/log4j-defaults-repl.properties
To adjust logging level use sc.setLogLevel("INFO")
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (OpenJDK 64-Bit Server VM, Java 1.8.0_292)
Type in expressions to have them evaluated.
Type :help for more information.
21/07/22 13:59:19 WARN Utils: Your hostname, saji-VB resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interfac
e enp0s3)
21/07/22 13:59:19 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Spark context available as sc.
21/07/22 13:59:28 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
21/07/22 13:59:30 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
21/07/22 13:59:43 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so
 recording the schema version 1.2.0
21/07/22 13:59:44 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
21/07/22 13:59:53 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
21/07/22 13:59:54 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
21/07/22 14:00:14 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so
 recording the schema version 1.2.0
21/07/22 14:00:14 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
SQL context available as sqlContext.

scala>
```

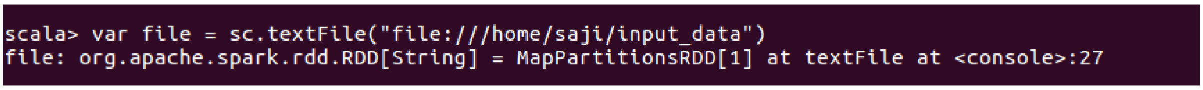
Figure 1 Start the spark shell

Word Count

Step 2: Create RDD from a file in HDFS,

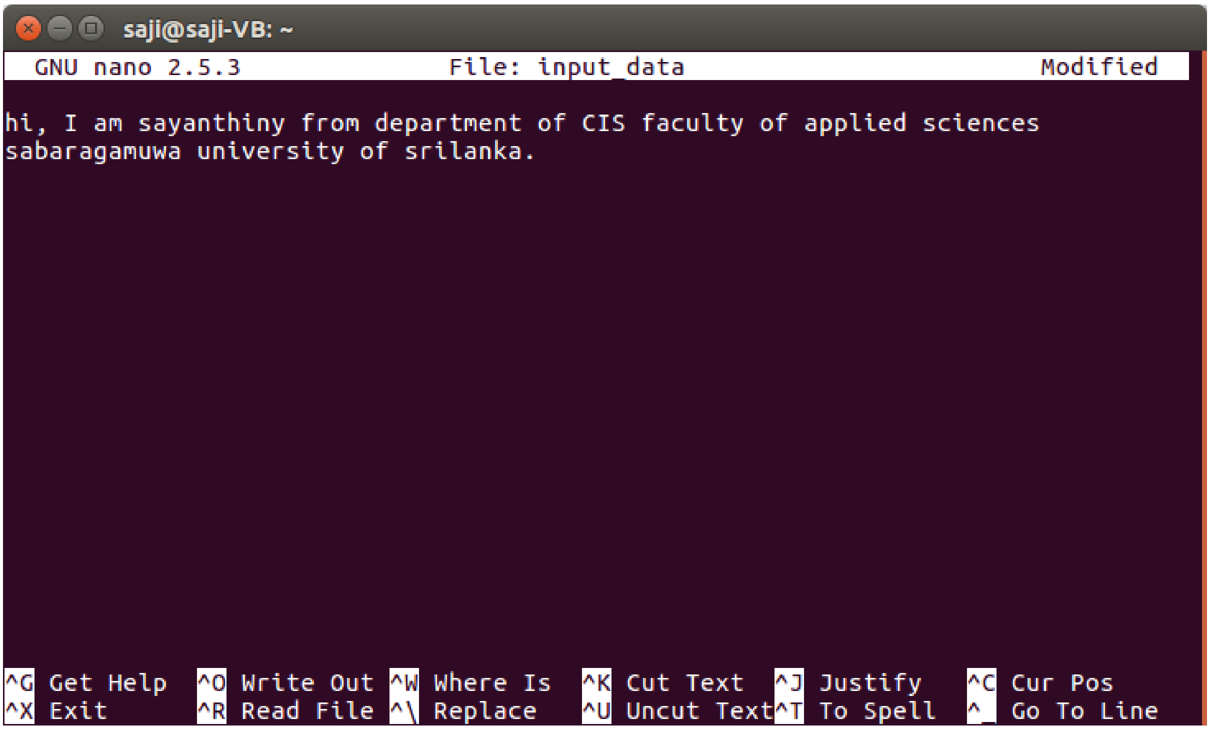
To read a text file from HDFS (or a local file system) and return it as a RDD of Strings, use the SparkContext object, which represents a connection to a Spark cluster and may be used to create RDDs, accumulators, and broadcast variables on that cluster.type the following command on spark-shell and press enter:

```
scala> var file = sc.textFile("file:///home/saji/input_data")
```



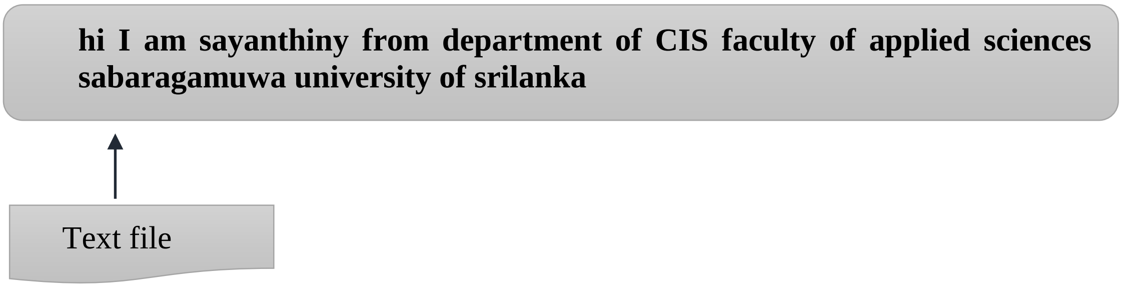
```
scala> var file = sc.textFile("file:///home/saji/input_data")
file: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[1] at textFile at <console>:27
```

Figure 2 Create RDD from a file in HDFS



```
  GNU nano 2.5.3              File: input_data                        Modified

hi, I am sayanthiny from department of CIS faculty of applied sciences
sabaragamuwa university of srilanka.

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

Figure 3 input data edit with nano



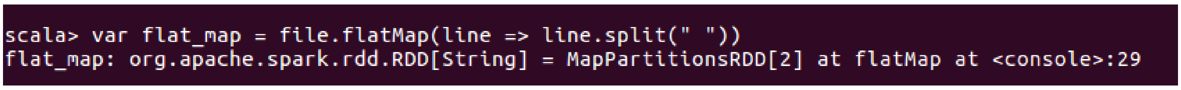
hi I am sayanthiny from department of CIS faculty of applied sciences sabaragamuwa university of srilanka

Figure 4 input text

Step 3: Convert record into words

two-layer map is created if only a map function is used to split the RDD of Strings.

```
scala> var flat_map = file.flatMap(line => line.split(" "))
```



```
scala> var flat_map = file.flatMap(line => line.split(" "))
flat_map: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at <console>:29
```

Figure 5 Convert record into words

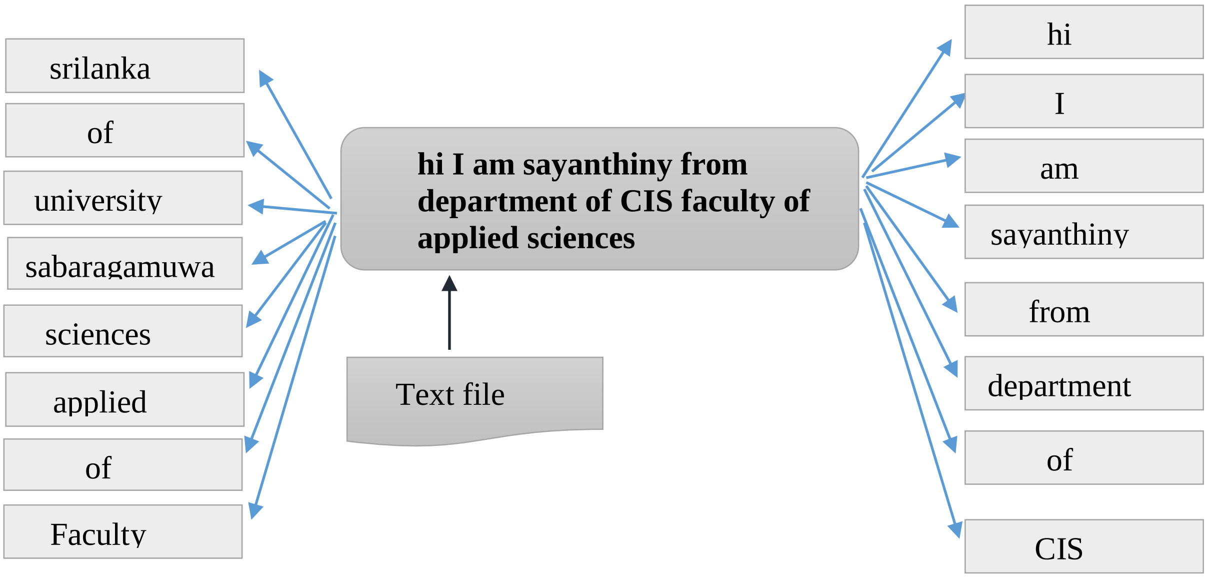
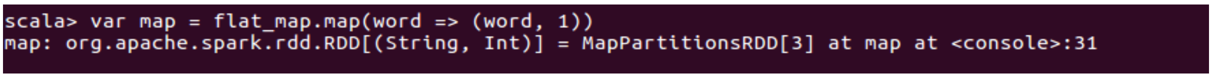


Figure 6 split text into words

Step 4 : Convert each word into key-value pair

Each element in the set should be mapped to a map, with only one occurrence of each element in the map. Use the following commands to convert word into key value pair.

```
scala> var map = flat_map.map(word => (word, 1))
```



```
scala> var map = flat_map.map(word => (word, 1))
map: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at <console>:31
```

Figure 7: perform map operation

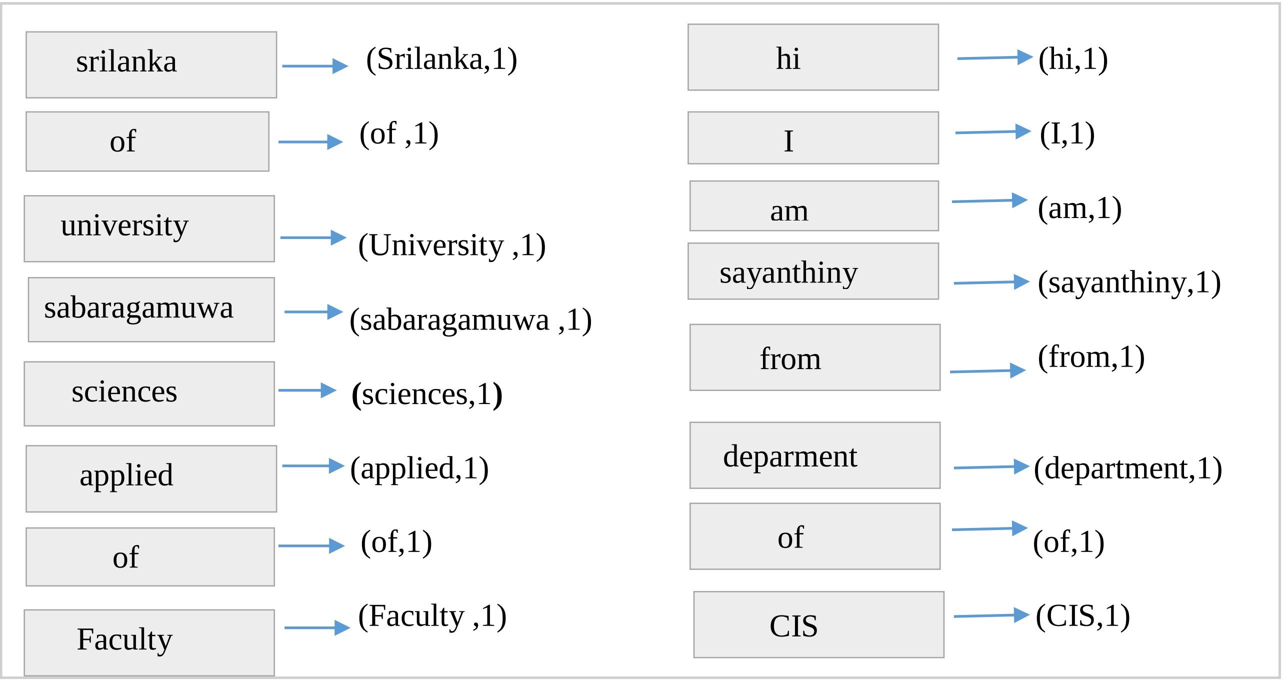
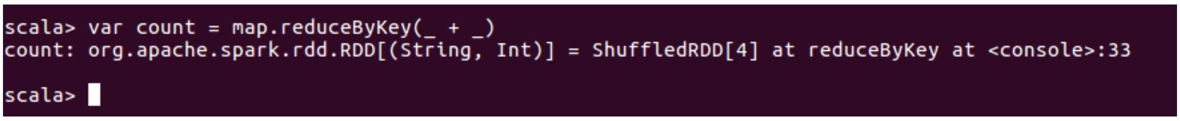


Figure 8 map function

Step 5 : Group By key and perform aggregation on each key:

All map items should be reduced by key and multiplied by the number of times they appear in the map.

```
scala> var count = map.reduceByKey(_ + _)
```



```
scala> var count = map.reduceByKey(_ + _)
count: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:33

scala> █
```

Figure 9: perform reduce operation

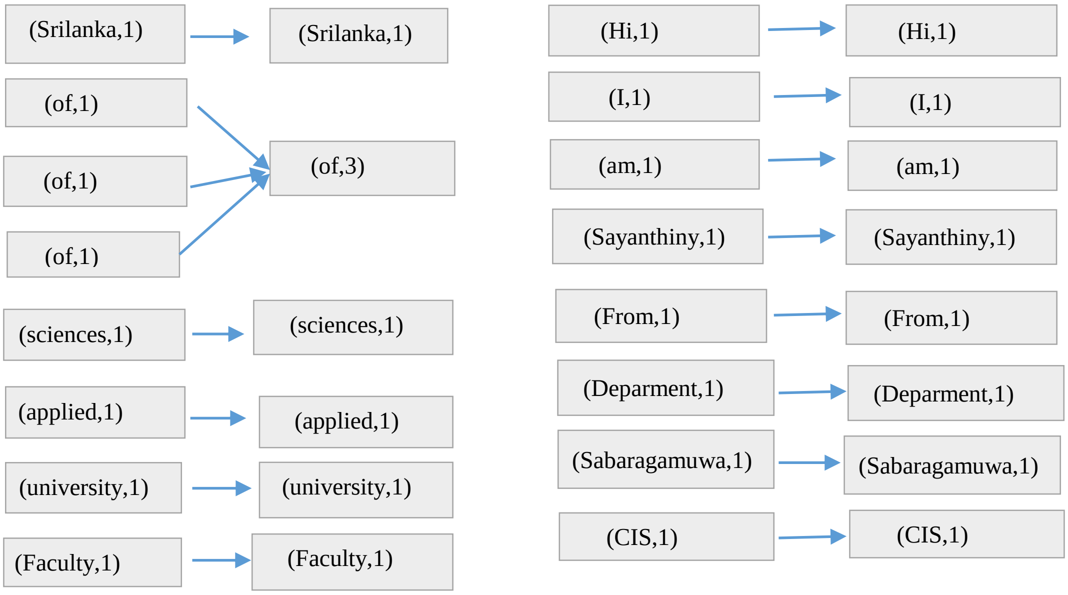
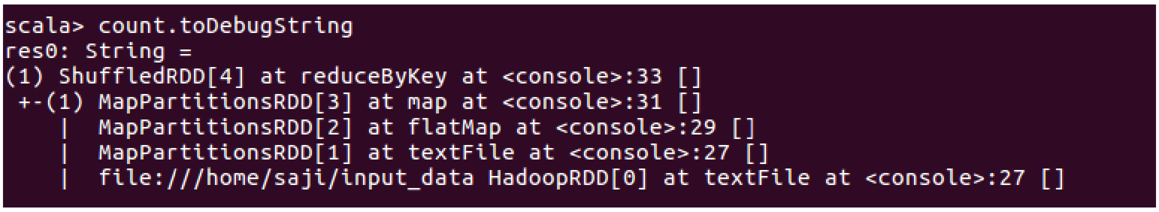


Figure 10 reduce function

Step 6: Description about Current RDD

If you want to know about current RDD, then use the following command. It will show you the description about current RDD and its dependencies for debugging.

```
scala> counts.toDebugString
```



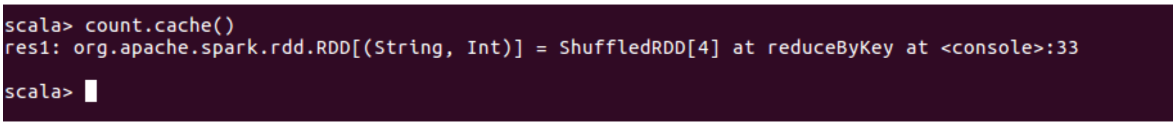
```
scala> count.toDebugString
res0: String =
(1) ShuffledRDD[4] at reduceByKey at <console>:33 []
 +-(1) MapPartitionsRDD[3] at map at <console>:31 []
    |  MapPartitionsRDD[2] at flatMap at <console>:29 []
    |  MapPartitionsRDD[1] at textFile at <console>:27 []
    |  file:///home/saji/input_data HadoopRDD[0] at textFile at <console>:27 []
```

Figure 11 Description about Current RDD

Step 7: Caching the Transformations

Use the following command to store the intermediate transformations in memory.

```
scala> counts.cache()
```



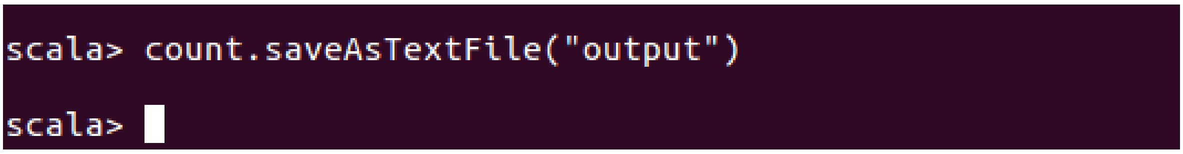
```
scala> count.cache()
res1: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:33

scala> 
```

Figure 12 store the intermediate transformations in memory

Step 8: Store all the transformations, results into a text file

```
scala> counts.saveAsTextFile("output")
```

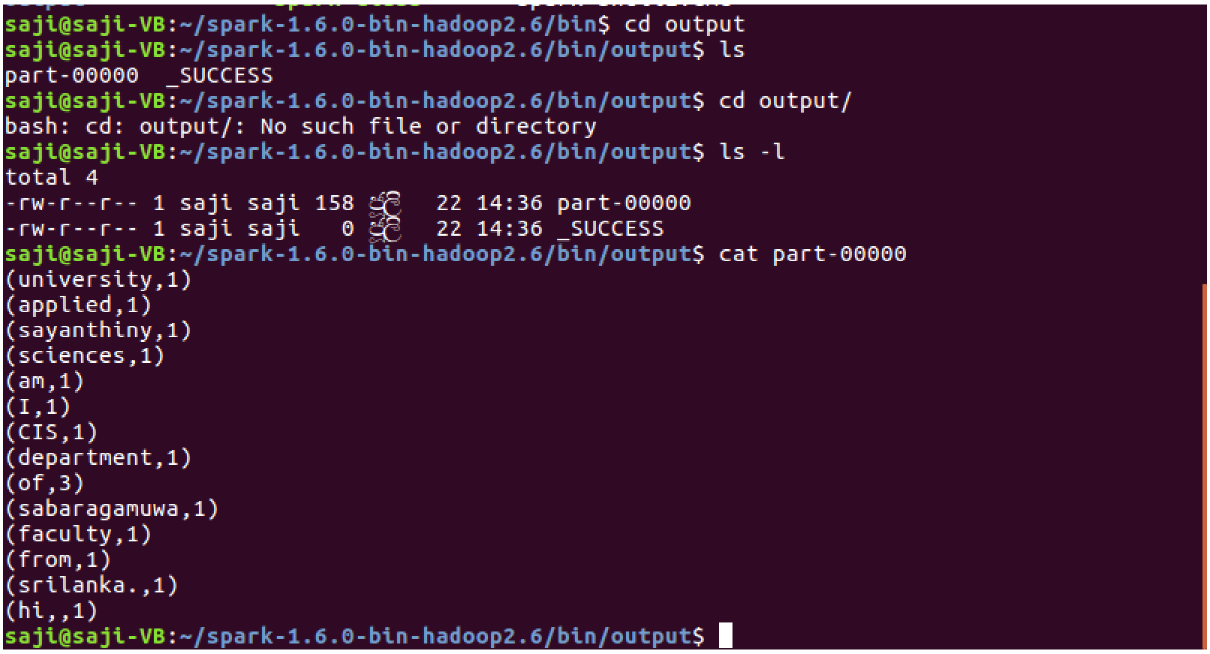
store all the transformations, results into a text file. Following command to save the output in a text file.



```
scala> count.saveAsTextFile("output")

scala> 
```

Figure 13 output

Step 9: Checking the Output



```
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin$ cd output
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin/output$ ls
part-00000  _SUCCESS
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin/output$ cd output/
bash: cd: output/: No such file or directory
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin/output$ ls -l
total 4
-rw-r--r-- 1 saji saji 158 [illegible]  22 14:36 part-00000
-rw-r--r-- 1 saji saji   0 [illegible]  22 14:36 _SUCCESS
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin/output$ cat part-00000
(university,1)
(applied,1)
(sayanthiny,1)
(sciences,1)
(am,1)
(I,1)
(CIS,1)
(department,1)
(of,3)
(sabaragamuwa,1)
(faculty,1)
(from,1)
(srilanka.,1)
(hi,,1)
saji@saji-VB:~/spark-1.6.0-bin-hadoop2.6/bin/output$ 
```

Figure 14 check the output