Spark Laboratory

This laboratory is dedicated to Spark. The main goal for this laboratory is to gain familiarity with the flexibility offered by Spark. The interested student can find the details of Spark:

- in the website: http://spark.apache.org/docs/latest/quick-start.html
- in the following book: "Learning Spark – Lightning-Fast Big Data Analysis" by Holden Karau, Andy Konwinski, Patrick Wendell, Matei Zaharia

Downloading a pre-compiled released version of Spark

For this laboratory we do not need the VM with Hadoop installed. We instead download a pre-compiled version that includes everything, and we will use the interactive shell to launch the jobs.
The only requirement is to have a Linux-based machine (it works also on Mac OS).

Go to http://spark.apache.org/downloads.html and

- Choose a Spark release (e.g., the latest release);
- Select the package type of "Pre-built for Hadoop 2.4 and later";
- Select "Direct download".

Once you have downloaded the compressed tar file (e.g., spark-2.1.0-bin-hadoop2.7.tgz), unpack it and move to the Spark directory. From there, you can launch the interactive shell.

```
tar -xf spark-2.1.0-bin-hadoop2.7.tgz
cd spark-2.1.0-bin-hadoop2.7/
./bin/spark-shell
```

The following figure shows a screenshot of the (Scala) shell. If you want to interact with the Python shell the command is "`bin/pyspark`".

```
[carra] spark-2.1.0-bin-hadoop2.7 $ ./bin/spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
17/01/19 16:22:34 WARN SparkContext: Support for Java 7 is deprecated as of Spark 2.0.0
2017-01-19 16:22:35.311 java[75660:1903] Unable to load realm info from SCDynamicStore
17/01/19 16:22:36 WARN NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
17/01/19 16:22:36 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
17/01/19 16:22:45 WARN ObjectStore: Failed to get database global_temp, returning
NoSuchObjectException
Spark context Web UI available at http://157.27.230.7:4040
Spark context available as 'sc' (master = local[*], app id = local-1484839357655).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.1.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_55)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Interacting with Spark: simple examples

The following examples use Scala: for the Python examples, please refer to the website or the book indicated at the beginning of the document.

The interactive shell contains a built-in Spark Context ("`sc`"): we will use the Spark Context to load the file into a RDD, then we will do some RDD transformations or we will apply some actions.
Let's start by loading the file "README.md" and count the number of lines and display the first 3 lines.

```
scala> val lines = sc.textFile("README.md")
lines: org.apache.spark.rdd.RDD[String] = README.md MapPartitionsRDD[1] at textFile
at <console>:24

scala> lines.count()
res0: Long = 104

scala> lines.take(3)
res2: Array[String] = Array(# Apache Spark, "", Spark is a fast and general cluster
computing system for Big Data. It provides)
```

Next, let's count the words:

```
scala> val words = lines.flatMap(lines => lines.split(" "))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at
<console>:26

scala> val wordskv = words.map(word => (word,1))
wordskv: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at map at
<console>:28

scala> val counts = wordskv.reduceByKey((a,b) => a + b)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at
<console>:30

scala> counts.collect()
res3: Array[(String, Int)] = Array((package,1), (this,1),
(Version"](http://spark.apache.org/docs/latest/building-spark.html#specifying-the-
hadoop-version),1), (Because,1), (Python,2),
(page](http://spark.apache.org/documentation.html).,1), (cluster.,1), (its,1),
([run,1), (general,3), (have,1), (pre-built,1), (YARN,,1),
([http://spark.apache.org/developer-tools.html](the,1), (changed,1), (locally,2),
(sc.parallelize(1,1), (only,1), (locally.,1), (several,1), (This,2), (basic,1),
(Configuration,1), (learning,,1), (documentation,3), (first,1), (graph,1), (Hive,2),
(info,1), (["Specifying,1), ("yarn",1), ([params]`.,1), ([project,1), (prefer,1),
(SparkPi,2), (<http://spark.apache.org/>,1), (engine,1), (version,1), (file,1),
(documentation,,1), (MASTER,1), (example,3), (["Parallel,1), (are...
scala>
scala> counts.saveAsTextFile("../data/wordcount")
```

The action `collect()` on a RDD shows the initial content of the RDD. We can save the result in a file with the following command

```
scala> counts.saveAsTextFile("./wordcount")
```

Exercise – Analysis of the tstat data

Consider the data and the exercises about tstat done with Pig. Re-do all the exercises with Spark.