Universität Freiburg
Institut für Informatik
Dr. Fang Wei-Kleiner
Victor Anthony Arrascue Ayala

Georges-Köhler Allee, Geb. 51
D-79110 Freiburg
fwei@informatik.uni-freiburg.de
arrascue@informatik.uni-freiburg.de

Lab Course: Distributed Computing Using Spark
Summer Term 2021

1. Experiment: Familiarizing with Tools, Loading Data, and Basic Analysis of Data

Deadline: 15.5.2021

Submission Guidelines:
For this experiment you need to submit a Jupyter Notebook (.ipynb).
Please follow the conventions for this kind of file(s). You will find them in ILIAS (*Submission_Conventions.pdf*).
Remember to use comments to explain your solutions. Not commented solutions will not be marked!
Upload your solution to ILIAS:
`https://ilias.uni-freiburg.de/goto.php?target=crs_2174945&client_id=unifreiburg`.

Exercise 1. 1 (Environment setup and dataset, 0 points)
In the hand-on session you had the opportunity to use some of the technologies we will use during the course of the lab:

- Docker: `www.docker.com` and `https://hub.docker.com/r/jupyter/all-spark-notebook/`
- Jupyter Notebook: `http://jupyter.org/`
- Python: `https://www.python.org/`
- Apache Spark: `https://spark.apache.org/`
- PySpark (Python API for Spark): `http://spark.apache.org/docs/2.2.0/api/python/pyspark.html`

Familiarize yourself with those tools. At the end of this sheet you will find some useful resources to learn more about them. In addition, familiarize yourself with the dataset "citeulike". This dataset is a collection of ratings from users given to scientific papers. In ILIAS you will find the file *citeulike_dbis_experiment_1.zip*. The compressed file contains the dataset and a *README.txt* file. Read it in order to get a better understanding of the content of each file. For the sake of clarity, we report here the files which are part of the dataset:

- *papers.csv*
- *users_libraries.txt*
- *stopwords_en.txt*

Exercise 1. 2 (Loading the dataset into an RDD, 10 points)
RDD and DataFrames are two different data models in Spark. RDDs are Spark's abstraction of distributed collections and they provide an API which allows one to work with them in a functional style. In this task you will learn how to create a particular kind of RDD, the so-called ***pair*** RDD. This kind of RDD allows one to organize the collection as key-value pairs.

Load the dataset carrying out the following steps:

a) *userRatingsRDD*: create a pair RDD from *user_libraries.txt* using the user hash as the key and the liked paper(s) (*paper_id*) as the value(s).
b) *paperTermsRDD* : create a pair RDD from *papers.csv* using the *paper_id* as the key and the words contained in the abstract as the value(s).

Note: read the data directly into RDD, i.e. do not build DataFrames or any other data models and then convert them to RDDs.

Exercise 1. 3 (Joining collections, 10 points)
Create a script that computes for each user the top-**10** most frequent words appearing in the papers she likes. Exclude the stop words listed in *stopwords_en.txt*.
Store the results into a file which contains in each line the user hash and the list of her retrieved words sorted by frequency (top 1 is the most frequent):
<user_hash>, top_1_word, top_2_word, top_3_word,..., top_10_word
Note 1: remember that a collection is not only distributed on a number of nodes but also replicated in order to support fault-tolerance. This behavior is not what one needs for the stop words file. Instead, one would like to have access to all stop words at each node so that each of them can use it (no distribution). This can be achieved by means of broadcast variable(s). Broadcast variables enables one to have *read-only* global access to a resource from each of the (in your case virtual) nodes. Broadcast variables can transfer the data in a more optimized way, however these should be only use if the data is small.
Note 2: expect a computation time of approx. 40 min when 2GB are dedicated to the computation.

Exercise 1. 4 (Basic Analysis for Recommender Systems, 10 points)
One of the initial tasks when designing a recommender system consists of carrying out some analysis on the data you have. The goal of this basic analysis is to gain an idea of the characteristics of the dataset. Using Spark's capabilities, retrieve the following information:

a) Number of (distinct) user, number of (distinct) items, and number of ratings
b) Min number of ratings a user has given
c) Max number of ratings a user has given
d) Average number of ratings of users
e) Standard deviation for ratings of users
f) Min number of ratings an item has received
g) Max number of ratings an item has received
h) Average number of ratings of items
i) Standard deviation for ratings of items

Exercise 1. 5 (Loading the dataset into Data Frames, 10 points)
In contrast to RDDs, DataFrames allow one to handle structured, distributed data in a table-like representation with named and typed columns. DataFrames are therefore applicable in any instance that requires manipulation of structured data.

Load the dataset into DataFrames leveraging the structure of the data. Choose a reasonable schema.

Exercise 1. 6 (Tasks on top of DataFrames, 10 points)
Solve the tasks 1.3 and 1.4 again using the DataFrames built in the previous task instead of RDDs. Use only DataFrame's functionalities, i.e. avoid converting your model to RDD and then using RDD's functionalities. For 1.3 you need to investigate the *Window Functions in Spark SQL*.
Moreover, record the execution times for each task and data model. Is there any noticeable performance difference between RDDs and DataFrames when executing the tasks? Justify your answer.

References:

1. *Jupyter Notebook:* `https://www.youtube.com/watch?v=HW29067qVWk`
2. *Jupyter Notebook's documentation:* `http://jupyter-notebook.readthedocs.io/en/latest/notebook.html`
3. *Introduction to Spark:* `http://researchcomputing.github.io/meetup_spring_2014/python/spark.html`
4. *Introduction to Pyspark:* `http://nbviewer.jupyter.org/github/jkthompson/pyspark-pictures/blob/master/pyspark-pictures.ipynb`