School of Electrical and Computer Engineering,
Technical University of Crete
Academic Year 2023-2024 (Spring Semester)
INF424 Functional Programming, Analytics and Applications

Instructor: N. Giatrakos, Lab Tutor: C. Tsinaraki

Project Topic: Functional Programming Analytics and Applications with Scala on Apache Spark

Weighs 5/10 units of the INF424 Course
Project Teams of up to 2 students

Preamble on Implementation Specifications

In the scope of this project, we are going to use functional programming to perform Big Data analytics over a state-of-the-art Big Data Platform, namely Apache Spark.

- In all analytics scenarios in the scope of this project you must use the entire datasets. Solutions that work with subsets but not with the entire dataset in each scenario will not be evaluated at all.
- The implementation of both analytics scenarios in this project should be in Scala using the higher order (transformation) functions provided by Apache Spark. Implementations in other languages or using anything outside Apache Spark will have your project desk rejected.
- The data in both analytics scenarios must be first properly loaded to HDFS. The required analytics as well as any preprocessing, data cleaning or postprocessing on the data should be performed in Apache Spark using Scala. No side projects outside Scala@Spark are allowed.
- In each analytics scenario, the type of functional data structures (Spark RDDs or Spark Dataframes) which should be used is explicitly mentioned. Follow these instructions. You are not allowed to use mere spark.sql.
- In each scenario, your code must be able to run on the SoftNet Cluster. You should contact the Lab Tutor: C. Tsinaraki to create an account to the SoftNet cluster and acquire connection instructions. You should use the SoftNet cluster ONLY AFTER you have developed your code and you want to finally confirm its execution on a real cluster. You cannot use the SoftNet cluster for development purposes. Abusing the SoftNet cluster may have your account banned. Projects that run locally on your laptop but cannot run on the actual cluster have a penalty of -75% of the units assigned to the corresponding analytics scenario.

Application Scenario 1: Reuters News Stories Analysis using Spark RDDs (3 Units)

In the context of this application scenario, we will use the files available in the Reuters Dataset, included in the current project's folder. This is a semi-structured dataset, therefore one has to use Spark RDDs to proceed with the analysis. Those of you who attended INF211 in the previous semester are already aware of the structure and properties of this dataset.

This dataset includes a set of files with information about **news stories (documents)** which are assigned to specific **categories** each. Each document also includes a set of **terms** of interest to analyze in conjunction with existing news categories.

More specifically, the purpose of the analysis is, by compiling the information in the various records, to extract the relevance of each term to each category. To quantify this degree of relevance, we will use the Jaccard Index as a measure.

The files that concern us are the following:

The ASCII file rcv1-v2.topics.qrels describes, for each document, the categories to which it has been assigned to. Each pair <category, documentid> is represented in a record and there are 2.606.875 records, each having the form:

<category name> <document id>1 (the "1" at the end of each record can be ignored)

For example, we have the records:

E11 2286 1

ECAT 2286 1

MCAT 2286 1

C24 2287 1

CCAT 2287 1

These show that the document with documentid = 2286 belongs to the E11, ECAT, MCAT categories and the document with documentid =2287 belongs to categories C24 and CCAT.

5 ASCII files, with a total of 804,414 entries:

- lyrl2004_vectors_test_pt0.dat
- lyrl2004_vectors_test_pt1.dat
- lyrl2004_vectors_test_pt2.dat
- lyrl2004_vectors_test_pt3.dat
- lyrl2004_vectors_train.dat

Each record in these files refers to a particular document and includes the list of terms that it contains in the form <did> [<term id>: <weight>]+ where

<did>: Reuters-assigned document id.

<term id>: A positive integer as the unique identifier of each term (term id). Term ids are between 1 and 47.236.

<weight>: We are not interested in these fields in our scenario. You can safely ignore them.

For example, the record:

2286 864: 0.0497399253756197 1523: 0.044664135988103 1681: 0.0673871572152868 ....

says that the document with did = 2286 contains terms with term ids 864, 1523, 1681, ...

An ASCII file stem.termid.idf.map.txt which maps each term id to the stem/token it describes. There are 47.236 records in the file, one for each term being monitored. Entries are in the form of:

<stem> <termid> <idf> where:

<stem>: the root (etymologically) of the term

<term id>: the unique identifier of this term

<idf>: It can be ignored in our scenario.

For example, records

map 25376 6.41892286389214

reduce 34542 2.83102755083624

match termid = 25376 with "map" and termid = 34542 with the term "reduce" (stem).

Given the above, your Scala@Apache Spark code should do the following:

1. Calculate |DOC (C) | for each and every category.
2. Calculate | DOC (T) | for each and every term.
3. Calculate | DOC (T) ∩ DOC (C) |.

For each <T, C>, i.e., (term, category), pair we want to extract the following information in an output file stored in HDFS:

<category name> ;<stemid>;<JaccardIndex>

where

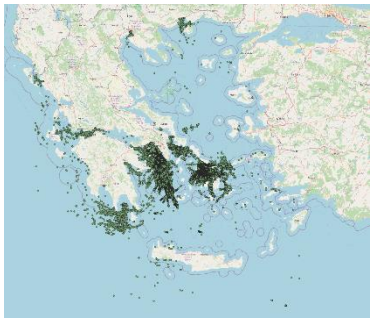
$$\text{JaccardIndex}(T, C) = \frac{|DOC(T) \cap DOC(C)|}{|DOC(T) \cup DOC(C)|}$$

Tip: It holds that $|DOC(T) \cup DOC(C)| = |DOC(T)| + |DOC(C)| - |DOC(T) \cap DOC(C)|$

Application Scenario 2:
Vessel Trajectory Analytics in the Aegean Sea using Spark Dataframes (20%)

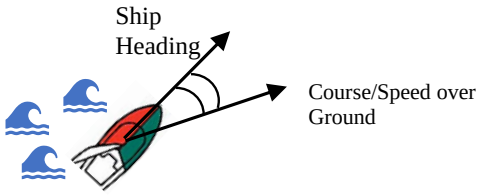
In this application scenario we are going to perform basic analysis on vessel position tracking data collected from a set of stations located in Piraeus and Syros Island. All the necessary information is included in the nmea_aegean.logs file accompanying this project announcement.

The dataset has been made available to us by the https://smartmove.aegean.gr/ laboratory (many thanks!), for the purposes of the course and only within the scope of the course. Please do not share this dataset further.



The nmea_aegean.logs file is a file with schema as described below:

- timestamp: the timestamp of the acquired vessel position
- station: the id of the station which acquired the corresponding vessel position
- mmsi: maritime mobile service identity, consider this as the vessel Id
- longitude: geographic longitude of each timestamped vessel position
- latitude: geographic latitude of each timestamped vessel position
- Speed Over Ground (SOG): is the speed of the vessel relative to the surface of the earth
- Course Over Ground (COG): is the actual direction of progress of a vessel, between two points, with respect to the surface of the earth
- Heading: is the compass direction in which the vessel bow is pointed



- Status:
  - 0 = underway using engine
  - 1 = at anchor
  - 2 = not under command
  - 3 = restricted maneuverability
  - 4 = constrained by her draught
  - 5 = moored
  - 6 = aground
  - 7 = engaged in fishing
  - 8 = underway sailing
  - 9 & 10 = reserved for future amendment
  - 11 = power-driven vessel towing astern
  - 12 = power-driven vessel pushing ahead or towing alongside
  - 13 = reserved for future use
  - 14 =AIS-SART Active (Search and Rescue Transmitter), AIS-MOB (Man Overboard), AIS-EPIRB (Emergency Position Indicating Radio Beacon):
  - 15 = undefined = default (also used by AIS-SART, MOB-AIS, and EPIRB-AIS under test).

Your code should provide answers to the following queries (results dumped as separate files in HDFS and printed on the driver screen):

Q1: What is the number of tracked vessel positions per station per day?

Q2: What is the vessel id with the highest number of tracked positions?

Q3: What is the average SOG of vessels that appear in both station 8006 and station 10003 in the same day.

Q4: What is the average Abs (Heading – COG) per station?

Q5: What are the Top-3 most frequent vessel statuses?

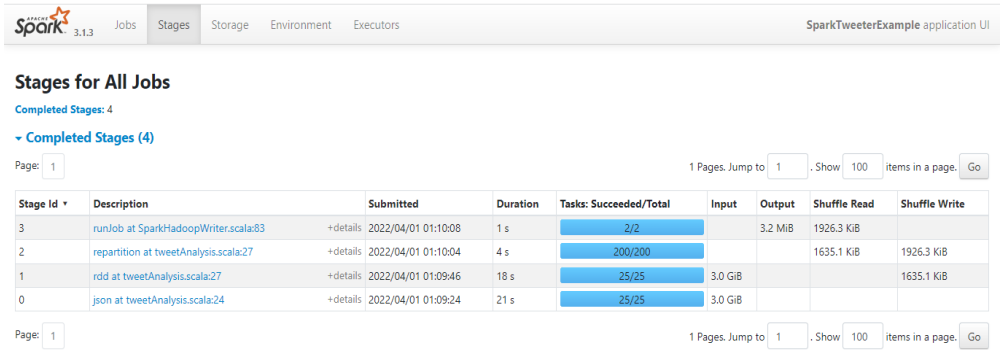
Deliverables

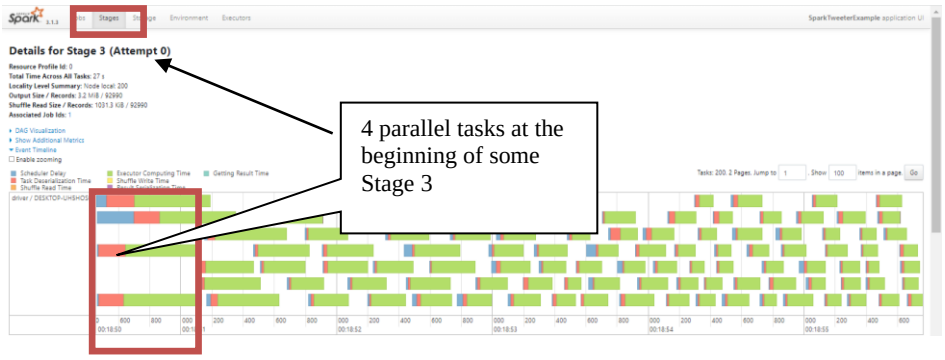
A. Implementation Report and Commentary

A report that will describe the logic of the program and algorithms you developed per Application Scenario.

- Comment on your code line by line, justifying the choices you made in order to demonstrate your understanding of the context of your implementation.
- Use figures and descriptions to show that you actually comprehend the way your code was executed on Spark, commenting on:
    - The jobs that Spark created per analytics scenario. Why do we have this number of jobs? How were they decided by Spark?
    - The stages of each job. Why the stages were split like this? Make references to parts of your code.
    - The tasks and the achieved degree of parallelism. What is the degree of parallelism? How were tasks decided by Spark?

Here are indicative samples of figures you should include in your report and use in your commentary. You can extract similar ones even from your execution locally at your laptop and comment on that according to the above requirements.





B. Code

Code for Application Scenario 1

Code for Application Scenario 2

C. Sample output

Provide a few lines (e.g. 50 lines) of each output file from your analytics results in both scenarios. For each analytics outcome you should include a separate file.

Project Logistics

- **Number of Project Group Members**: up to 2 students
- **Deadline**: 21/06/2024 by midnight. No extension will be provided under no circumstances.
- **Submission**: In a .zip file via eclass. Your submission must explicitly mention for each team member their student id and full name.
- **Date of Oral Examination**: the exact dates will be announced after your project delivery, but oral exams will take place during the last week of lectures. All project team members will be examined in the implementation of both Application Scenarios. You will have to execute a live demo of your code on your laptop AND on the SoftNet Cluster. The oral exam is mandatory for your projects to get evaluated.
- The grade of the project holds only for the exam period of June/July and September 2024.

Contact

Clarification requests should be addressed as follows:

- To (include both emails): ngiatrakos at tuc dot gr; ctsinaraki at tuc dot gr
- Email Subject: ProjectINF424 – FullName – StudentID using your academic email account

Academic Integrity Issues

Your code and reports will be checked using plagiarism detection software. Dealing with cases of plagiarism will be strict, and all involved project teams and individuals will have their projects zeroed out. Additionally, correct implementation by team members who are unable to answer questions during the oral examination will not be graded. Finally, the confirmed use of generative AI tools for the project code will also result in a zero score.

Enjoy!