UNIVERSITY OF AMSTERDAM

Universiteit van Amsterdam
BSc Informatica

Distributed and Parallel programming

Assignment for the module Big Data platform: docker containers /Spark

This assignment is a gentle introduction to Spark Big Data platform covered during the lectures. The aim is to cover both the virtualization/containerization and Spark. The focus of the assignment is to learn how to set up a **Spark** cluster and submit jobs to the cluster while Spark is deployed in Docker containers. **The assignment does not aim to achieve any performance using Spark, the assignment just gives you a 101 introduction to using Spark**.

Objectives of this assignment:

- **Part 1:** Create a docker container and deploy a spark-distribution
- **Part 2:** Use a REST API to submit jobs to spark cluster

Learning outcomes

1. Learn to create docker container
2. Learn to submit jobs to Spark cluster using the command line `spark-submit`, pyspak, and via a REST API

1. Software needed to do the assignment

For this assignment, the students need three software packages: pyspark3.0.0, spark3, python3.9, java18

Note this assignment was tested on
- `MAC M1 (OSx 12.4 Monterey)`
- `Debian 5.10 VM`

2. The Assignment

Create a spark cluster composed of two spark nodes, one node running the spark-master, the second node running one spark-worker. Because, we do not have access to clusters which support virtualisation (DAS is an HPC cluster without support for docker virtualisation), the cluster will be running on your local desktop/laptop inside two docker containers. The containers will be orchestrated using the `docker-compose` tool.

Part 1: submit jobs to the Spark-cluster

TODO

- **Step_0**: Preparation for the assignment
  a. Quick start with SPARK [1]
  b. Learn how write docker files [2]
  c. Learn how to write docker-compose files [3]

- **Step_1**: for this assignment we suggest following the procedure as proposed in the Bitnami repository; in this repo you will find all the needed information to create a container with Apache-spark [4]. At the end of this step, you will learn how to create a container and deploy an application in the container in this case Apache-spark, to complete the first part of the assignment you need to perform the following tasks:
  1. Adapt the docker file `docker.yaml` [5] to create a Bitnami container
  2. Run the Bitnami container and test if it is up: (1) Check if the SparkUI is running,
  3. Using the command line `spark-submit` [6] [7], submit a spark-job to your spark cluster to run the PI example – the PI example jars are part of the Spark distribution and can be found in the example directory of any Spark distribution
  4. Write a small client application using `pyspark` [8]. To submit, run the PI example.

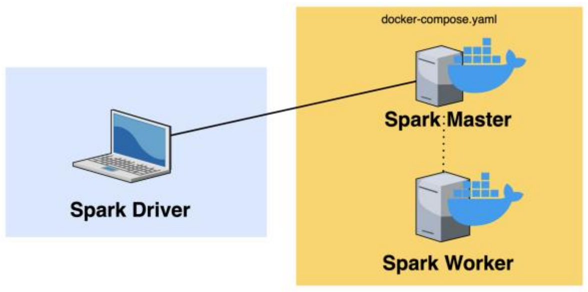

Figure 1: Cluster to setup in Part1 of the Assignment

[1] https://spark.apache.org/docs/latest/quick-start.html
[2] https://docs.docker.com/engine/reference/builder/
[3] https://docs.docker.com/compose/gettingstarted/
[4] https://github.com/bitnami/bitnami-docker-spark
[5] https://github.com/bitnami/bitnami-docker-spark/blob/master/docker-compose.yml
[6] https://sparkbyexamples.com/spark/spark-submit-command/
[7] https://spark.apache.org/docs/latest/submitting-applications.html
[8] https://spark.apache.org/docs/latest/api/python/

Part 2: submit jobs to the Spark-cluster via a REST API

In 2016 the Livy project [9] was developed to submit of Spark jobs from web/mobile apps (no Spark client needed) which enables multiple users to interact with the Spark cluster concurrently and reliably. Using Livy server you will be able to submit spark job to your spark cluster through a REST API.

TODO

1. Deploy the Levi server in the Bitnami container you have created in part1[10],
2. Extended the docker-compose yaml file to create a three containers: one running the Livy server and two containers running the Spark-server and the Spark-work[11]
3. Test that cluster is up via Spark UI, and the Livy UI
4. Write a small client application using to submit run the PI example [12]

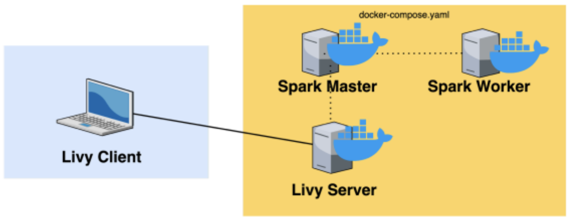

Figure 2: Spark cluster to setup in the part2 of the assignment

References (extra)

1. Deep Dive into Monitoring Spark Applications Using Web UI and SparkListeners (Jacek Laskowski) https://www.youtube.com/watch?v=mVP9sZ6K__Y
2. https://data-flair.training/blogs/learn-apache-spark-sparkcontext/

[9] https://livy.apache.org/
[10] https://livy.apache.org/download/
[11] https://github.com/bitnami/bitnami-docker-spark/blob/master/3.1/debian-11/docker-compose.yml
[12] Resource scheduling https://livy.apache.org/examples/

Grading

- if a student creates the containers from scratch starting from the docker-files as described in the assignment, the student can get the full grade (Grade up 10 depending on the number of correct answers).
- if the student uses pre-build container with spark, the student can get the only grade up 9 depending on the number of correct answers.
- If the student complete only one part of the assignment, the student gets a pass (Grade 5.5)

Note: Pre-build images will be provided by the TA at the request of the students.