Big Data Processing and Analysis
Term Project

Feyza Şahin - 18011019
Cihat İslam Dede - 19011047

General Information

This project is based on the analysis of the "GoodReads Books" dataset which was created by the information collected from the Goodreads website where book lovers make their own virtual library. The users can list the books which they already read and they also can rate them. In this dataset, we have some features about the books such as author, publish year, page number, language, rating scores, review counts etc. We will implement the project by using some of these columns to retrieve information about the readers and their booklists.

As an environment, we use Apache Hadoop which has the open-source software tools that enable a network of numerous computers to address problems involving enormous volumes of data and processing. It offers a software framework for the MapReduce programming model-based distributed storage and processing of massive data.

Use Case Scenario

In today's businesses, data size has been increasing daily at exponential rates. Thanks to the developing technology, it is feasible to evaluate the data and derive the answers for what is sought. The processing and analysis of data changes when the data is too large to implement all the processes in a single machine. Large volumes of data are analyzed using big data analytics to find undiscovered patterns, correlations, and other insights. In our case, we implement an application which is run on a Hadoop ecosystem that handles the large volume data by running the codes in a distributed system with the map-reduce method.

Due to the increasing readership and the books published, the data on the Goodreads website is increasing much faster. Thus, the required time for the statistical data analysis on this dataset is growing. We aim to prevent any time-consuming implementation caused by the dataset's large size, through a multi-node cluster environment.

Technical Challenges and Solutions

Initially, we tried to set up the Hadoop environment by creating virtual machines on VMWare, but these organizations were too repetitive and difficult to manage. Then we discovered that this could be done with Docker containers. We created our own Dockerfile and created the Hadoop environment on Docker. Thanks to the containers in Docker we can package up an application with all of its necessary parts, such as libraries, and dependencies.

Explanation of the Implementation

As mentioned earlier, we used Apache Hadoop which provides us to run our code and process the data in a distributed system with its major components as MapReduce, YARN and the HDFS. We wrote the map-reduce classes and functions in Java programming language by managing the dependencies via Maven in IntelliJ as IDE. As a graphical user interface, we used Python with the tkinter library.

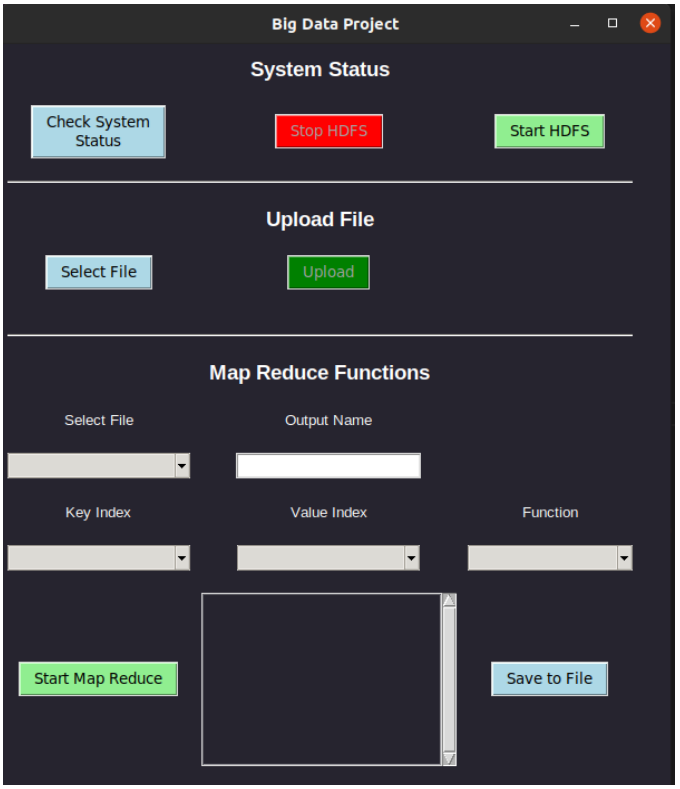

Figure 1- Graphical User Interface (GUI)

The distributed file system known as HDFS manages massive datasets on commodity hardware.

We used Docker for OS-level virtualization. Docker's platform as a service offerings distribute software in packages known as containers. Containers may be used as very portable, modular virtual computers using Docker. We have flexibility with those containers since we can develop, deploy, copy, and transfer them from one environment to another,

which aids in the cloud optimization of your programs. Using the kiwenlau repository, we created our own Dockerfile and bash script files to make our work easier.

As stated in the project proposal, we implemented the functions for the desired columns below:

1. Average
2. Minimum and maximum
3. Count
4. Sum
5. Standard deviation

Performance Evaluation

1) We evaluated the operations in terms of time on both single-node and multi node clusters (#node=2) to see the efficiency of a multi-node cluster. We tested the process on small, big, and huge datasets for the 'average' function.

2) We evaluated the operations in terms of time on a multi-node cluster to see the efficiency of a multi-node cluster for different sized files.

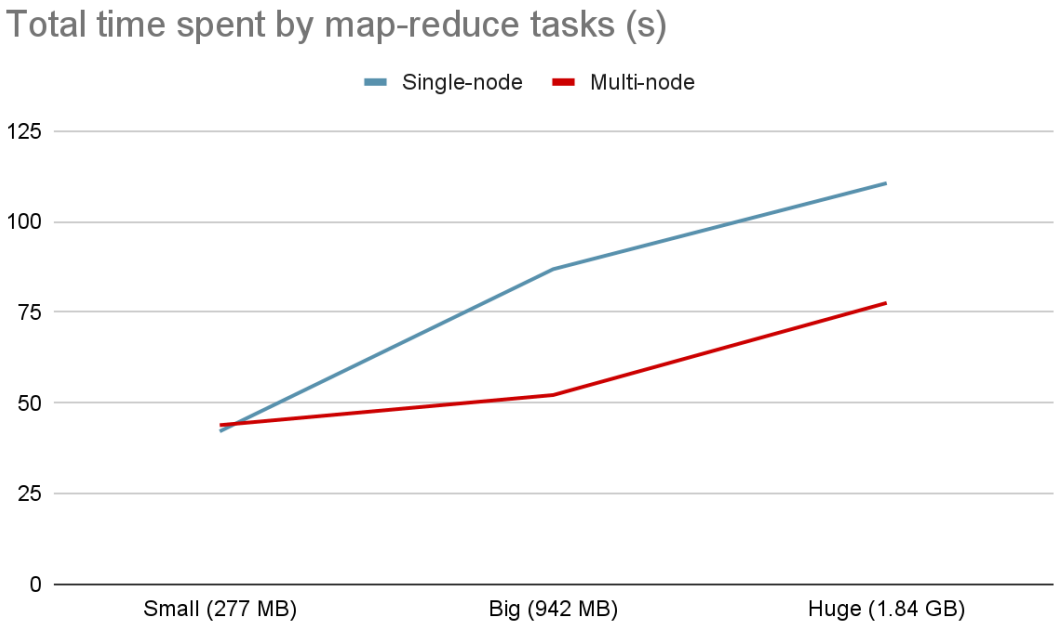

Figure 2- Single and Multi Node File Size Comparison (lower better)

Experience and Discussion

Through this project, we understood the distributed system proposed for handling massive data. We have written and learned how the system maps the data and then reduces it to implement the process faster, in the code steps. Thanks to the distributed system, we can split the data into the pieces for processing each at different nodes in the cluster and then bring the results together coming from these nodes.

References

- Goodreads Book Datasets With User Rating 2M | Kaggle
- Run Hadoop Custer within Docker Containers
- Essential Docker Commands
- Installation Of Multi-Node Hadoop 3.2.1 Cluster on a Single Machine
- Hadoop Mapreduce
- https://github.com/a-Imantha/average-calculation-map-reduce