Università degli Studi di Verona

Authorship Attribution

Big Data project report

Davide Bianchi VR424505
Matteo Danzi VR424987

A.A. 2018/2019

Contents

1 Introduction

The project aim was to design a tool which could establish the authorship of a manuscript by using specific criteria described later. The used architecture is based on Hadoop, a distributed filesystem simulator, running in a docker container. The requirements of the project imposed the use of MapReduce programming model as seen in Laboratory Lessons of Big Data course.

2 Background and System Description

2.1 Cloudera Docker and system setup

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is an executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

Docker containers images become containers when they run on Docker Engine. They isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

The Cloudera[2] Docker image used in this project contains an Hadoop Distributed File System (HDFS) partition. Cloudera provides a scalable, flexible, integrated platform that makes it easy to manage rapidly increasing volumes and varieties of data in an enterprise.

For the installation of Docker, the Cloudera Docker image and the execution of jobs using jar file it has been used the instructions of the course. The project is written using Java version `11.0.4`. For the code compilation we used IntelliJ version `2019.2.4` compiling using compatibility mode with version `1.7`.

The whole project relies on HDFS (*Hadoop Distributed FileSystem*)[1], used to store files across server collections, and MapReduce, a programming framework providing tools in order to simplify the analysis of large datasets. The Hadoop version used in this project is `Hadoop 2.6.0-cdh5.7.0`.
The distributed filesystem consists on two basic services:

- *NameNode*: is a single instance service, running on a master node, which is responsible for files and directory mantainance within the filesystem itself;
- *DataNode*: running on multiple nodes (many instances are allowed), are supposed to retrieve the application data when they are told by the NameNode.

MapReduce is a programming paradigm which works on parallel platforms in order to process in the best possible way large amounts of data. The client has to implement a map function, which is supposed to take as input raw data and process them into intermediate *key-value* pairs, and the reduce function, which aim is to regroup key-value pairs relying on the key and generate key-value output pairs. The mapreduce model is explained more in detail in the next section.

2.2 Map Reduce Framework

This project uses the MapReduce Programming Model. It allows expressing distributed computations on massive amounts of data. Considering it as an execution framework it is designed for large scale-data processing, and to be executed on clusters of commodity hardware.

3 Project Workflow

The first step the project is to analyze an amount of manuscripts, extracting relevant information from them in order to create a "dictionary" of known authors. Starting from these data, the program should take unknown manuscripts as input and establish a possible author, comparing the extracted data with the "dictionary" previously created.
This Programming Model consists of two main actions:

- **Map**: application of a function to every object in a list. Each object (e.g.: document) is independent.
  - Order is not important
  - Maps can be done in parallel since every object has no relation to each other
  - The function produces an intermediate result

  Formally the *map* function can be described with the following notation:

  $$map : (key_1, value_1) \rightarrow list(key_2, value_2)$$

- **Reduce**: combining the intermediate results generated by Maps to produce a final result.

4 Project Workflow

The first step of the project is to analyze an amount of manuscripts, extracting relevant information from them in order to create a "dictionary" of known authors. Starting from these data, the program should take unknown manuscripts as input and establish the authorship by the comparing them with the "dictionary" previously created. All the manuscripts are `.txt` files taken from the Project Gutenberg[3] online library. The Project Workflow is the following:

1. Parsing all the files using Hadoop Map Reduce job:
   - Mapper performs the counting of all the specified language parts in each file. These parts are ArrayList instances. The Mapper associates each language part with an integer flag set to 1. The collected data are explained more in detail in the next sections.
   - Reducer performs the counting of language parts by putting together data taken from the Mappers. The reducer output
2. Calculating the frequencies of language parts for each text.
3.

4.1 Map Reduce Job

4.2 Frequency Calculations

4.3 Similarity Analysis

References

Online Sources

[1] M. Afzali, N. Singh, and S. Kumar. “Hadoop-MapReduce: A platform for mining large datasets”. In: *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. 2016, pp. 1856–1860.

[2] Apache Software Foundation Cloudera Inc. *Cloudera Official Website*. 2019. URL: https://it.cloudera.com/.

[3] Michael Hart. *Free eBooks - Project Gutenberg Official Website*. 2019. URL: https://www.gutenberg.org/.