UNIVERSITY OF PISA

Artificial Intelligence and Data Engineering

Cloud Computing

Hadoop Letter Frequency

Authors: Gemelli M. Namaki D. Nocella F.

Academic Year 2023/2024

Contents

1 Introduction 2

2 Algorithm Design 2

3 Results 2

4 Conclusions 2

Abstract

This report describes the project developed for Cloud Computing exam at the University of Pisa. It explores the algorithm design and the discussion about obtained results.

Introduction

The objective of this project is to implement a data processing pipeline that can handle substantial data sets, ensuring efficient computation and meaningful data insights. By exploiting the MapReduce paradigm, data processing tasks is split into two main functions: the Mapper and the Reducer. The Mapper function processes and filters the input data, emitting key-value pairs, while the Reducer function aggregates and processes these pairs to produce the final output.
The project was developed using the Hadoop framework, which provides an open-source implementation of the MapReduce paradigm. Hadoop allows for the distributed processing of large data sets across clusters of computers using simple programming models. The Hadoop ecosystem also includes other tools, such as HDFS (Hadoop Distributed File System) for distributed storage, and YARN (Yet Another Resource Negotiator) for cluster resource management.

Given a text document as input, it is aimed to extract the frequency of each letter composing such document. In Order to achieve this, two diffent jobs are required: the first job is responsible for counting the number of letters in the document, while the second job is responsible for evaluating the frequency of each letter.

Algorithm Design

In order to reduce the amount of data to be transfered to the reducer (or reducers) two different strategies are used:

- **Combiner**: a combiner is mini-reducer that aggregates the output of the mapper before sending it to the reducer. It performs its operations on the same node where the mapper is running.
- **In-Mapping combiner**: this is a design pattern that allows to perform some aggregation inside the mapper itself. In details, it performs the combiner operation directly in the mapper, before emitting the key-value pair.

Results

Conclusions