|  |  |
| --- | --- |
|  | |
| Programming for Big Data –Assignment 1  Hadoop/Map-Reduce | |
|  | |
| Ciaran Finnegan – TU060 (Part Time) – Second Year  PROG9813: 2022/2023  MSc in Computer Science (Data Science)  Student No : D21124026  26/2/2023 |  |
|  |  |

Table of Contents

1 Introduction 3

1.1 Purpose of Report 3

2 Loading Data Into HDFS 4

2.1 Setting Up Shared Folders 4

2.2 Language Files Source (Books) 4

2.3 Transferring and Loading Data Files 4

3 Map-Reduce Process Design 6

3.1 High Level Overview 6

3.2 Explanation of Map-Reduce Workflow 7

3.3 Key Design Decisions and Assumptions 9

4 Map-Reduce: Java Code Walkthrough 10

4.1 Driver 10

4.2 Job 1 10

4.3 Job 2 10

5 Map-Reduce: HDFS Outputs 11

5.1 Job 1 11

5.2 Job 2 11

6 Python: Graph Analysis of Outputs 12

6.1 Set Up Excel From HSFs Output 12

6.2 Python Jupyter Notebook 12

6.3 Analysis Commentary 12

# Introduction

## Purpose of Report

This report collates all the delivery requirements for Assignment 1 in the Programming for Big Data module (PROG9813: 2022-23).

The primary content of the report covers;

* How language data (book files) were sourced and then loaded into a HDFS data store on a Linux VM running Hadoop (the VM provided during lecture 1).
* How the Map-Reduce workflow is constructed, to ingest the language files stored in HDFS and ultimately construct a breakdown of the average character frequency in each language.
* The Java source written to build the Map-Reduce processes, and how the features offered by Map-Reduce were employed to resolve the assignment challenge.
* Design assumptions and decisions made during the implementation of this PBD assignment.
* Python source code and output (via a Jupyter Notebook) of the analysis on the output from the Map-Reduce process.

Supporting source code and HDFS outputs accompany this report as part of the overall Brightspace submission.

# Loading Data Into HDFS

## Setting Up Shared Folders

The VM provided during lectures was used as the Hadoop environment for this assignment.

Following lecturer advice, a shared folder on the host machine was set up.

<Image – setting on VM>

## Language Files Source (Books)

Six books were selected from the <http://www.gutenberg.org> website;

* Two in Spanish.
* Two in German.
* Two in Italian

.

<Image – Txr files in Shared directory>

## Transferring and Loading Data Files

The filenames of each book were altered in the transfer process from host machine to VM to add a suffix indicating language.

<image – shell file bookcopy.sh>

The renaming was done as part of the design assumptions covered in section 3.3 of this report.

Setting up a book directory and loading the files into HDFS was done through the following Linux commands;

<image – shell file bookcopy.sh>

The Hadoop interface confirmed that the language book files were successfully loaded.

<Image – After data loaded in HDFS>

A quick inspection (example below) showed that the data was intact and ready for processing.

# Map-Reduce Process Design

## High Level Overview

This diagram displays the high-level Map-Reduce process built and deployed for the assignment.

<image - workdlow>

The following sections of this report elaborate on the individual Map/Reduce processes.

## Explanation of Map-Reduce Workflow

### Driver Process

There is one Driver process for the entire assignment. Key steps in the process are;

* One…
* Two…
* Three…

### Job 1: Mapper/Combiner Process

Key steps in the first (Job1) Mapper process are;

* One…
* Two…
* Three…

..

### Job 1: Partitioner Process

Key steps in the Partitioner process are;

* One…
* Two…
* Three…

<Explanation for Partitioner process…>

### Job 1: Reducer Process

Key steps in the first (Job1) Reducer processes are;

* One…
* Two…
* Three…

### Job 2: Mapper Process

Key steps in the second (Job2) Mapper process are;

* One…
* Two…
* Three…

..

### Job 2: Reducer Process

Key steps in the second (Job2) Reducer process are;

* One…
* Two…
* Three…

## Key Design Decisions and Assumptions

In the design and implementation of PBD CA1 the following were the key design assumptions and decisions;

* One…
* Two…
* Three…

# Map-Reduce: Java Code Walkthrough

## Driver

The…

<image>

The…

## Job 1

### The Mapper…

<image>

### The Partitioner…

<image>

### The Reducer…

<image>

## Job 2

### The Mapper…

<image>

### The Reducer…

<image>

# Map-Reduce: HDFS Outputs

## Job 1

### Output 1

The…

<image>

The…

### Output 2

The…

<image>

The…

## Job 2

The…

<image>

The…

# Python: Graph Analysis of Outputs

## Set Up Excel From HSFs Output

The final output from the Map-Reducer process is downloaded on the VM and then copied to the host machine.

The text file is loaded into Excel and the inbuilt wizard creates an \*.xlsx file output wit the data in three columns (no headers).

<image>

All \*.xlsx files are then copied to the same local directory from which the Python Jupyter Notebook written for the assignment is stored.

## Python Jupyter Notebook

The Python code reads all locally stored \*.xlsx files (all of which are assumed to be in the HDFS output format) and stores this language data into separate Panda dataframes.

The notebook generates a character frequency graph for each language dataframe.

Below is the Jupyter Notebook code with generate output included.

<image>

## Analysis Commentary

Commentary on each language graph is included in the output in Section 3 of the Python Jupyter Notebook.

A brief description is provided on how the distribution of characters for each language varies from English. The average letter frequency for English given in the brief for PBD Assignment 1 is used as the basis for this comparison.

Given that the assignment brief shows a letter frequency based on the Oxford English dictionary, it would be necessary to greatly increase the size of input for ‘foreign’ languages in our VM to give a more accurate comparison. Resource limitation for this CA meant that it was necessary to restrict the size of the Map-Reduce input.