CS6220 – Fall 2021

Assignment 1

Name: Junyan Mao

GTID: 903343678

Disclaimer:

- The code for the word count program is adopted from Apache's official MapReduce tutorial found in the following link: https://hadoop.apache.org/docs/stable/hadoopmapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
- This has been approved by Professor Liu in this Piazza post: https://piazza.com/class/ksp6ajd2q5h69g?cid=26

Dataset Sources:

- files in datasets/20/
  - fetched from project Gutenberg's website https://www.gutenberg.org/
- files in datasets/alice/
  - Alice's Adventure in Wonderlands fetched from project Gutenberg's website
- files in datasets/hello_world/
  - manually created simple "Hello World" text
- files in datasets/mark_twain/
  - all of Mark Twain's works, fetched from project Gutenberg's website
- files in datasets/pride/
  - Pride and Prejudice fetched from project Gutenberg's website
- files that are too big to include:
  - Amazon reviews
  - downloaded from https://www.kaggle.com/bittlingmayer/amazonreviews
  - used test set as amazon_small.txt, used training set as amazon_big.txt

Hardware Configuration:

- WSL (Windows Subsystem for Linux)
- Ubuntu 20.04 LTS
- Java OpenJDK 11.0.11
- Hadoop 3.3.1

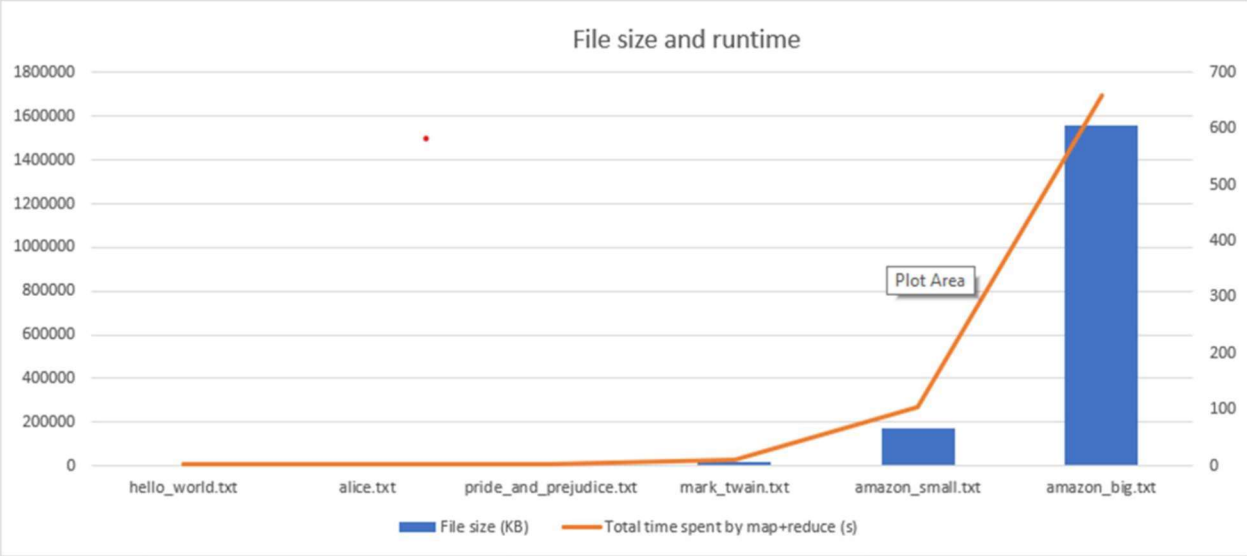
MapReduce Job Log Screenshots:

- Located in screenshots/ folder

Summary:

In this project, I used Hadoop MapReduce to run the WordCount program on datasets of different sized datasets to observe the scaling effect of Hadoop. I picked datasets ranging from several KBs to over 1 GB, which represent the common workload for WordCount. At first, I had trouble setting up Hadoop 3.3.1 on my MacOS machine, and even when I was able to run it, it couldn't print out the metrics properly. I used my Windows machine running WSL for the rest of this assignment and everything worked fine. For the count-top-100-words program, I used another MapReduce job in which the Mapper reverts the previous <key, value> pair of <word, count> into <count, word>, and the Reducer selects the top 100 words.

Runtime Analysis:



Hadoop MapReduce can handle smaller files very well. "hello_world.txt", "alice.txt", and "pride_and_prejudice.txt" take about the same time to finish (~3 seconds), though each of them are multiple times bigger than the previous one. I think we can assume that this is mostly the overhead of setting up whole Hadoop pipeline to prepare to run the MapReduce job, the actual job itself only takes a fractional of the time.

When it comes to bigger files, Hadoop also does a good job. From "mark_twain.txt" to "amazon_small.txt", the file size 11x'd and the runtime also 11x'd. But from "amazon_small.txt" to "amazon_big.txt", the file size 9x'd while the runtime only 6.35x'd. We can see that the scaling effect is becoming more obvious when we reach bigger file sizes.