Assignment 5

COMP3358 Distributed and Parallel Computing

Assignment 5

- **Hadoop and Spark**, both developed by the Apache Software Foundation, are widely used open-source frameworks for big data architectures. Both Hadoop and Spark enables big data processing tasks to be split into smaller tasks. The small tasks are performed in parallel by using an algorithm (i.e., MapReduce), and are then distributed across a Hadoop cluster.
- Spark tends to perform faster than Hadoop and it uses random access memory (RAM) to cache and process data instead of a file system in Hadoop. This enables Spark to handle use cases that Hadoop cannot.
- **In this assignment**, you will run both Hadoop and Spark on your own computer:
  - **Task 1:** preprocess an input dataset using **Hadoop**
  - **Task 2 and Task 3:** analyze the preprocessed dataset using **Spark**

Setup Hadoop

- Because Hadoop is open source, you can download and install it (see the Hadoop webpage) on your own computer!
- Hadoop Single Node Installation Reference: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html
- The `conf/slaves` file specifies the hostnames or IP addresses of all the worker nodes. By default, it only contains localhost.
- Run the example WordCount application: https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html#Source_Code

Exercise (Hadoop)

- **Task 1: Preprocess data.** Process the provided Trending Youtube Video Statistics dataset (*USvideos.csv*). Strip the *publish_time* in the dataset using Hadoop to leave only a specific part (the date) of the *publish_time*.
  - Example input: 2017-11-13T17:13:01.000Z
  - Example output: 2017-11-13
  - The preprocessed *USvideos.csv* is used as the input for the following two tasks.

Setup Spark

- **Apache Spark** is an open-source unified analytics engine for large-scale data processing. Spark provides an interface for programming clusters with implicit data parallelism and fault tolerance.
- **Download Spark:** https://spark.apache.org/downloads.html
- **Learn more about Spark:** https://spark.apache.org/examples.html
- You need to analyze the preprocessed *USvideos.csv*.
- Complete the following two tasks:
  - **Task 2:** Rank the most popular channels by aggregating their total views and ordering them accordingly.
  - **Task 3:** Rank the time periods with the highest number of videos: aggregate the number of videos by *publish_time* date, ordering them accordingly.

Setup pseudo-distributed Spark (cont.)

- **Run a Spark cluster on your machine**
  - Start the master node and one worker node with Spark's standalone mode (Spark Standalone Mode).
  - After starting the master node, you can check out master's web UI at http://localhost:8080 know the current setup
- **Run the example application with Spark** https://spark.apache.org/docs/latest/submitting-applications.html

Exercise (Spark)

- **Task 2: Rank the most popular channels by aggregating their total number of views and ordering them accordingly.** Aggregate the number of ***views*** for each *channel_title* in the statistics file, then rank them according to the total number of ***views***. The output should be the top ten *channel_titles* and the number of times their video have been viewed.
  - Example output: ("Peter Nicholls",902840) ("Peter Parker Tube",780538)
- **Task 3: Rank the time periods with the highest number of videos: aggregate the number of videos by publish_time date** (preprocessed by Hadoop)**, ordering them accordingly.** Aggregate the number of **videos** by their publish date, then rank them accordingly. The output should list the top ten dates along with the total number of videos released on each date.
  - Example output: (2017-11-13, 672) (2017-11-27, 267)
  - **Note: The example output is provided for formatting reference and does not represent the actual answer.**

Submission

- Submit all your **source file(s) and a document**. The document should contain the screenshots of the running program and the output results.