Big Data Analysis on Million Song Dataset

Yiding Chang, Kexuan Huang, Yifan Shen, Qinghang Wu

UNIVERSITY OF MICHIGAN
JOINT INSTITUTE
交大密西根学院

Introduction

To build a successful music platform, this project aims at developing an efficient music recommendation system with Hadoop, Drill, and Spark. Our goals are as follows.

- Use Drill to conduct basic analysis on the million song dataset.
- Use parallel Breadth First Search (BFS) to calculate the similarities and differences between artists.

Proposed Work Flow

In order to efficiently conduct the analysis, we propose to follow the below work flow with three milestones.

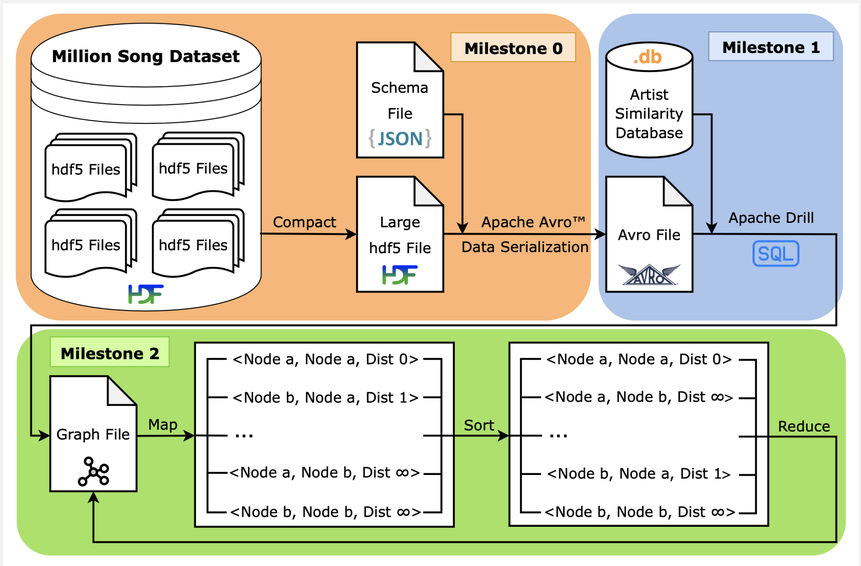

Figure 1: Work flow of the project

Basic Data Analysis

To test on the compacted dataset, we use drill to query one of the sub dataset and retrieve the following information:

- The oldest song is 96 years old, while the youngest is 12 years old.
- Hottest song that is the shortest and shows highest energy with lowest tempo is 'Immigrant Song (Album Version)'.
- 'Greatest Hits' has the most tracks, which is, 21.
- Name of the band who recorded the longest song is 'UFO'.

Parallel BFS Algorithm

For BFS with MapReduce, we used the following algorithm and implemented it in Python,

1. Initialize graph file with target artists in the format of 'Node, Distance, Neighbors'
2. Mapper: Generate neighbors, Node, Distance $+1$ if not itself
3. Reducer: Merge the same neighbor, keep distance minimum

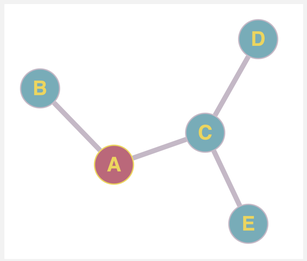

Figure 2: Initialize Graph File

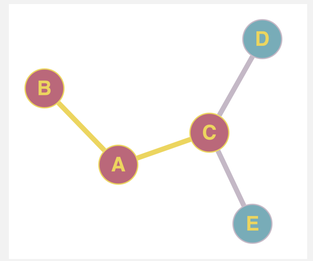

Figure 3: Graph after 1 MapReduce Iteration

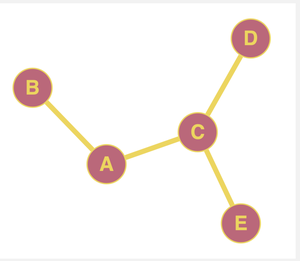

Figure 4: Graph after 2 MapReduce Iterations

For BFS with Spark, we implemented it using Python with 'PySpark' with the following steps:

1. Convert data into RDD map
2. Using Spark 'sortByKey()' to sort RDD aggregated by node index and then combining the neighbours
3. Using Spark 'reduce()' to pick the minimum distance of different neighbours towards the central node

The data size we used is around 200GB. The server we used is SJTU cluster with three machines with CPU of Dual-core Intel Xeon Processor (Skylake, IBRS) and memory of 4GB.

Conclusion

With big data technology, we have successfully implemented parallel BFS algorithms in both MapReduce and Spark to determine the similarities and differences between the artists. To summarize,

- Conducting basic queries on the dataset can provide lots of insights about the music industry.
- When running on the whole 200 GB dataset, on average, MapReduce finishes the task around 250s, while Spark finishes the task around 45s. Therefore, we conclude Spark is 5.5 times faster than MapReduce, so it would be a better option for the music platform.