A7 Airline Routing

Karthik Chandranna
chandranna.k@husky.neu.edu

Naveen Aswathanarayana
aswathanarayana.n@husky.neu.edu

Ruinan Hu
hu.ru@husky.neu.edu

Sujith Narayan
rudrapatnaprakash.s@husky.neu.edu

March 20 2016

DESCRIPTION

The aim of this assignment is to propose two-hop routes for a given origin, destination and a date. The possibility that a proposed two-hop route leads to a missed connection and the total duration of all the proposed connections should be minimized. A connection is any pair of flight F and G of the same carrier such as F.Destination = G.Origin and the scheduled departure of G is <= 1 hour and >= 30 minutes after the scheduled arrival of F. A connection is missed when the actual arrival of F < 30 minutes before the actual departure of G. When a proposed two-hop route is a missed connection, a penalty of 100 hours is applied. The output of the program is the total duration of all the proposed two-hop routes after applying a 100 hour penalty.

IMPLEMENTATION DETAILS

Design

The solution involves two MapReduce jobs and a Spark SQL program. The two MR jobs run on the given train and test datasets respectively and generate details of all the existing connections. The Spark SQL program runs on this intermediate dataset and the provided requests data to choose the connections having the least duration and also the least chance of a missed connection.

The key design idea is to have the first MR Job operate on the train (history) dataset and calculate the missed connection percentage for each intermediate city. The second MR Job will operate on the test data to generate the list of all possible connections. The Spark SQL program tries to avoid intermediate cities that have a high missed connection percentage, thereby minimizing the chances of incurring a 100 hour penalty.

First MapReduce Job

The first MapReduce job operates on the train dataset which is a history file having Airline details for 3 years from 2002 to 2005. For each row in the dataset, an AirlineParser object is created.

The Mapper job writes to the context twice. Once with the key as origin and the value as flight details, and then with the key as destination and value as flight details.

The reducer job iterates over the flight details and outputs the ratio of missed connections to connections for each intermediate city.

Second MapReduce Job

The second MapReduce job operates on the test dataset, which is a test file having airline details for the year 2004. For each row in the data set, an AirlineParser object is created. The Mapper job writes to the Mapper twice.

The Mapper job writes to the context twice. Once with the key as the combination of carrier and origin, the value as flight details, and then with the key as combination of carrier and destination.

The Reducer job iterates over the flight details and outputs the flight connections for each date.

Spark SQL

The Spark SQL program provides the best connection for each request based on missed connection ratio and duration of two-hop routes. Spark SQL is used to achieve joins across tables. We start by defining the schema and loading the historical and the test data sets. The history and test datasets are joined based on intermediate city to get the missed connection ratio. Temporary tables are created for request and connections, and these tables are joined based on date, origin and destination. Aggregate functions from the Spark Dataframe API are applied to pick the best connection for each request based on missed connection ratio and duration. We then calculate the total duration of the proposed connections after applying a 100 hour penalty for each missed connection.

Results

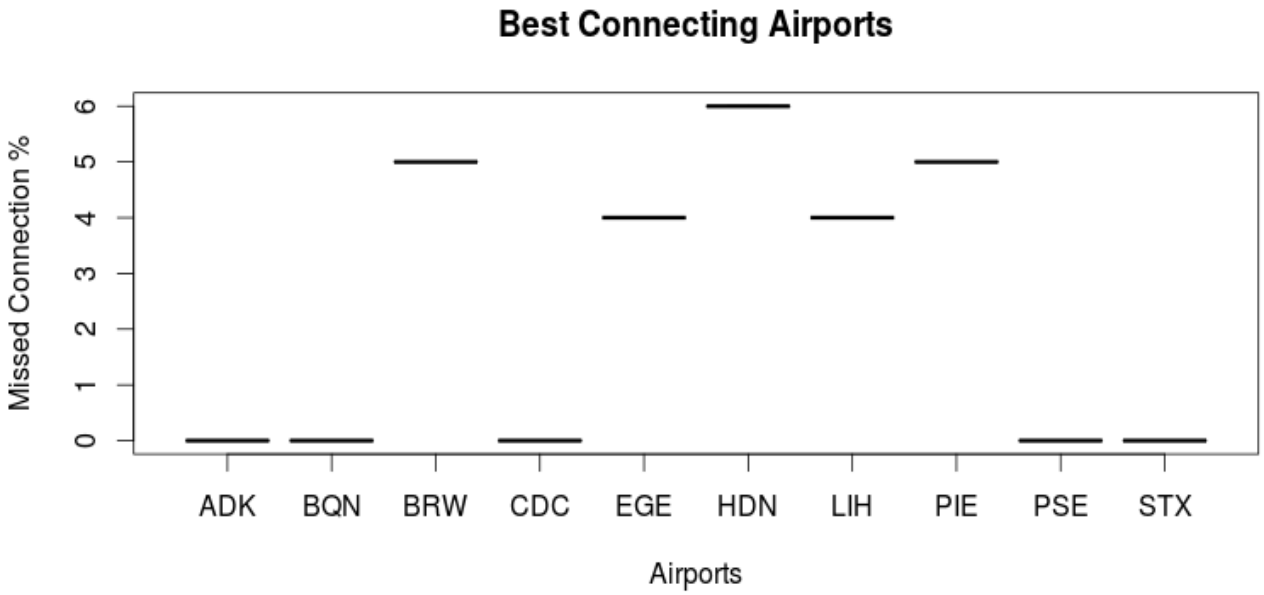
Travel duration (in hours): 32599

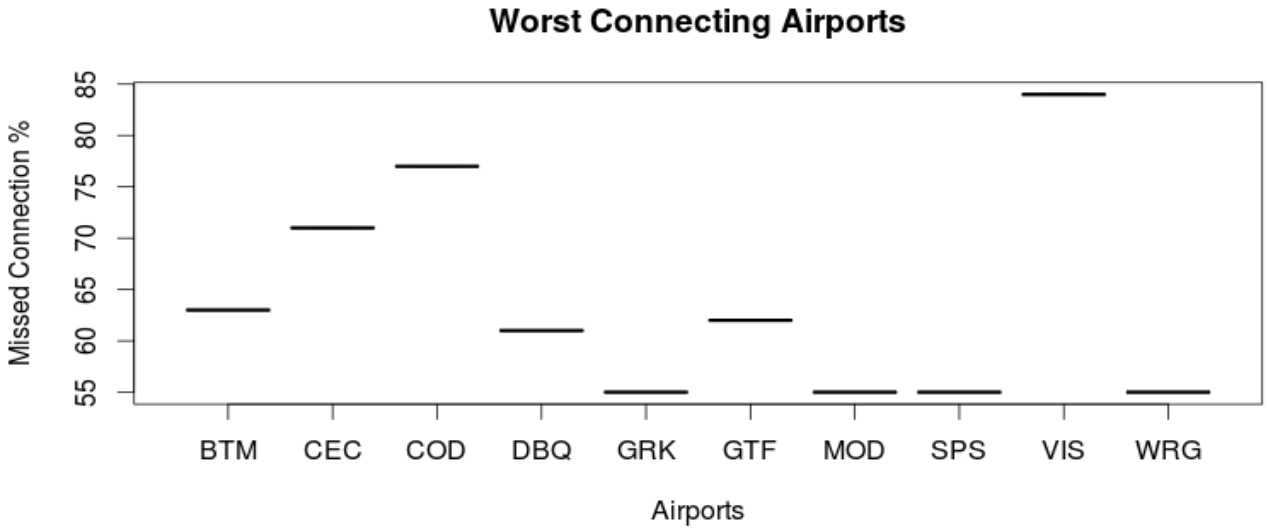
Number of missed connections: 525

Total duration after 100 hour penalty for each missed connection (in hours): 85099

Execution Time

Machine Configuration	Job Type	Time in Minutes
AWS – 1 Master 2 slave, m3.xLarge	Hadoop	7
	Spark SQL	12
AWS- 1 Master 4 Slaves, m3.xLarge	Hadoop	5
	Spark SQL	9
AWS- 1 Master 9 Slaves, m3.xLarge	Hadoop	4
	Spark SQL	8





Work Load Distribution

2 Map Reduce Jobs – Ruinan, Naveen

Spark SQL – Sujith, Karthik, Ruinan

Report and Code Documentation – Naveen, Karthik, Sujith