Spark Assignment Documentation

Spark Version 1.6.2 and Scala Version 2.10 is used for the development of this project.

Table of Contents:

Problem 1

Main class can be found: *src/main/scala/geosearch/Search.scala*

This Spark application is written in Scala to map given latitude longitude values to nearest latitude longitude values in a given set using functional programming simulating a map side join of two data sets.

This takes two sets of data as input. A **master** set of unique id, latitude and longitude values (which are considered constant), and a **search** set of latitude longitudes which are to be searched and mapped to this master set. This is expected to change on every run of the ETL/ join code. **This application is using broadcasting of a sorted index to reduce the need for a cross product using a map side join.**

Assumption:

- The master data set (id1, lat, long) can be cached in memory, thus not very large. This dataset is used as index. This data set is already sorted (or can be sorted using the code provided) based on latitudes and then optionally on longitudes.
- The master data set has all unique coordinates.

Mapping logic:

The two sets of coordinates are mapped based on the distance calculated by using the **Haversine formula**. The max limits considered as near can be specified while using the utility:

- maxDist - maximum distance that is considered as near (in kilometers) [used 50 here] *{note: taken from internet}*
- latLimit, lonLimit - To increase efficiency, the area in which the actual distances are calculated is narrowed down by using these limits.

  Eg: 0.5 latLimit means the distance of points beyond searchLat +- 0.5 are ignored, since 0.5 degree means 55.5 Kilometers approx, the mapped lat lon values would not lie beyond this range. Similarly for longitude, till 83 deg latitude the values for 49 Kilometers is 3.9 degree *{note: these values are taken from internet}*

Further Optimization Scope:

The logic implemented in this project, for optimization first goes through the broadcasted index data and filters out nearest matching Coordinate objects (Id, latitude and longitude value). Here **binary search can be leveraged to reduce complexity.**

Problem 2