COSC6339: Graph Analytics
Identifying the k Shortest Paths Among Every Pair of Vertices

1 Introduction

In this project you will create a Java/Vertica and Scala/Spark programs to find the shortest path between any pairs of vertices in a directed graph. A graph is defined as $G = \{V, E\}$ where V is a set of vertices or nodes and E is a set of edges of links between the nodes. Each edge has a distance v (weight/cost) associated with it.

2 Program and output specification

The main program should be called "topkpath", after being compiled, we must be able call the program by typing in:

Syntax:

```
topkpath k=<integer>
```

and the output shown on screen are the k longest paths between every pair of vertices.

3 Requirements

- The input will be a list of edges and corresponding cost in a csv file with three values on each line: i, j and v. We will provide the data set to test your program.
  - In Vertica: G will be represented by a table with three columns: i, j, and v.
  - In Spark: G will be stored as an RDD file cached across the cluster.
- Programming: you must use Java program for Vertica, and Scala for Spark. You are not allowed to export the data out of the database and process them in the Java/C++ and Scala programs. It must be done in-database, in other words, Vertica and Spark does the main job. ONLY the result is sent to the Java and Scala program to be displayed on the screen.
- The problem will be solved by calculating Transitive Closure of the graph G*, i.e. it is to iteratively calculate matrix multiplication as explained in [1].
- You can assume the maximum path length up to $k \leq 10$ edges.
- The program should not halt when encountering errors. It should just send a message to the log file and continue with the next line. The only error that is unrecoverable is a missing input file or a missing output file.

- Deliverables: source code and a README file showing how to compile and run the programs from the Unix terminal. Run the programs a few times and record differences between the two systems. Which one processes the query faster? You are expected to show and explain your program to the TA.

4 Extra credit

- 30 points for any teams showing the actual path. Note: the path should be stored INSIDE each system, only the final path to be retrieved from

References

[1] C. Ordonez, W. Cabrera, and A. Gurram. Comparing columnar, row and array dbmss to process recursive queries on graphs. *Information Systems*, 2016.