Efficient Graph-Based Solutions for the Traveling Salesman Problem with Apache Giraph and Hadoop

TRENT UNIVERSITY
CHALLENGE THE WAY YOU THINK

Introduction

In this study, we implement Travelling Salesman Problem (TSP) using Apache Giraph, a distributed graph processing platform, and Hadoop, a distributed computing platform. The TSP is a well-known optimization problem that involves finding the shortest possible route that visits a set of cities and returns to the starting point.

Methodology

We installed Java 1.8, Hadoop 1.2.1 and Apache Giraph 1.4. Then used GitHub repository for reference to implement the TSP algorithm. The input data was in the correct graph format and the Java code defined the logic for determining the shortest path.

Maven and Giraph JAR file helped in compiling the code and successfully build the Giraph model.

The results are successfully stored in an output directory.

Results

Giraph was able to successfully implement the TSP and execute it quickly. The following command line output displays the estimated distances between pairs of vertices/cities in the TSP graph using Apache Giraph.

Each line represents a vertex and the estimated distance between it and the rest of the graph's vertices. The line "1 0.0" represents, for example, the distance from vertex 1 to itself, which is 0.0. The line "2 10221.0" represents the 10221.0 distance between vertex 1 and vertex 2. Similarly, the line "3 1.0" depicts the 1.0 distance between vertex 1 and vertex 3. The remaining lines in the graph represent the distances between vertex 1 and other vertices.

Giraph v/s GraphX

The Facebook engineering team conducted a GraphX and Giraph performance comparison. On these platforms, several graph processing algorithms were performed and their runtime and scalability were evaluated. PageRank and Connected Components were among the algorithms investigated.

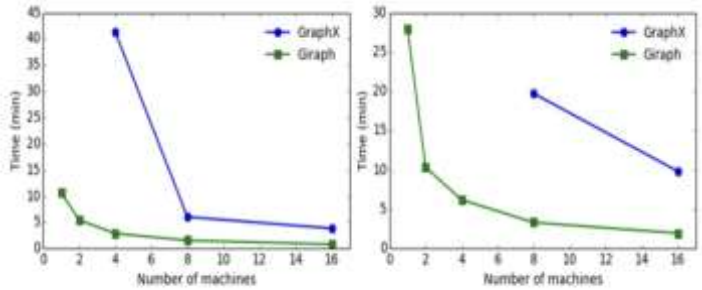

Fig. – Running Time of PageRank on the Twitter graph (left) vs UK Web Graph (right) as the number of machine varies

As shown above, Giraph can perform the PageRank algorithm on the Twitter graph 4.5 times quicker than GraphX with 16 workers and 4 times faster with eight workers. Overall, we observe that Giraph is better suited to handling production-scale workloads, whereas GraphX has a number of capabilities that make development easier.

Conclusion

This project helped us discover the potential of Apache Giraph by efficiently solving complicated problems like TSP. This demonstrated how effective Apache Giraph is and why corporations such as Facebook and LinkedIn choose it over other frameworks such as GraphX for social network research, connection recommendations, and so on.