**Metropolitan State University**

**ICS 440-01 Parallel and Distributed Algorithms**

**Summer 2019 - Assignment 4 Report**

This program seeks to achieve parallel execution of the Floyd-Warshall algorithm, an algorithm that determines the shortest path between all pairs of nodes in a directed graph (known as the “all-pairs shortest-path” algorithm). Design of this program follows a paradigm whereby the problem itself is partitioned appropriately, the partitions communicate as required, tasks are discovered and communication operations are agglomerated, and finally, the tasks are assigned a proper place in which to execute in the mapping stage.

**Partitioning**

An intuitive partitioning for this algorithm would be to split up the distance matrix into discrete parts, whereby each part can be computed independently of each other. The size of this partitioning may be dependent on other factors, such as communication between other sections of the overall partitioning. This represents a 2-D domain decomposition on the underlying data structure. Each individual part may be assigned to an execution process as a task of work to be completed, provided that dependencies between the parts does not impact the correctness of the algorithm.

**Communication**

A method of broadcasting information to all processes where dependencies are known to exist is required for correct and successful execution of the algorithm. In the distance matrix *D*, the only way that the *k*-th iteration of *Di,j* can be computed is if the (*k-1*)-th iteration’s computation is complete for the same *Di,j*. This is at least one (possibly the only) known dependency. Communication between the known, calculated parts of the *D*(*k-1*) matrix must be accomplished in order to calculate the unknown parts of the *D*(*k*) matrix. Therefore, each process requires row and column information from the *D*(*k-1*) matrix in order to continue with its own computation.

**Agglomeration**

**Mapping**

Each task is assigned its own process using a thread pool. The tasks and their execution are to be managed by the Java Executor framework using Futures. The computation of each part of the distance matrix *D*(*k*) will use Futures generated from the parts of the *D*(*k-1*) matrix, until the algorithm has completed its execution.

**Conclusion and Runtime Results**