Carl Smith

3313 Algorithms

11-26-14

Programming Project 4 Documentation

The problem the accompanying program seeks to solve deals with taking a v\*v adjacency matrix of a directed graph and outputting a v\*v matrix representing all pairs' shortest paths using Johnson's algorithm. In order to store the information, an array of vertices with links to neighboring (pointed toward) vertex information. Input is taken from the matrix.txt file where the array size, v, is the first line, and then every line afterword contains a row of v integers for the array. Edge weights to connected vertices are assumed to have values not of zero; zero weights have been considered other than directed edges to other vertices. Below details the key functions used in the code.

* displayMatrix: takes an array of vertices and displays them, showing each shortest path from each vertex to another, where rows represent source vertexes and columns represent end points. For each vertex that is unreachable from a source, a value of Integer.MIN\_VALUE will result in producing “inf” for the element in the console display. Values of zero indicate no cost in reaching another vertex (particularly the source vertex in itself).
* primeG & unprimeG: the former adds a new vertex to the front of the vertex array which connects toward all other points with zero-weighted edges; the latter simply removes the first element from the list.
* BellmanFord: takes G' as input along with its additional vertex 's' and returns a false if a negative edge cycle is found or a true after finding the shortest source paths to each vertex.
* Dijkstra: used on every vertex in G in order to determine the shortest paths for all source vertices to end-vertices.

1. Johnson: creates G', calls BellmanFord and returns a null on a false return and otherwise unprimes G'. The distances discovered by BellmanFord algorithm are then saved in array h and then edge weights in G are reweighted. Dijkstra's algorithm is then called for every vertex in G, after which all path values from that source vertex to every other are saved into the D array; the process is repeated until all vertices have been used as a source in Dijkstra's algorithm.