MST In a Directed Graph Name : Sasidhar Evuru

Abstract - The main aim of the project is to find the Minimum spanning tree in a directed Graph.

Problem Statement – Given a Directed Graph find the MST of the given graph and print the MST and also the traversal path if the number of edges is <= 50.

Implementation – The Algorithm which is discussed in the class is used. To implement the algorithm the following Data structures are used.

1. The Adjacency list for vertices is stored in the arralylist of linkedlists. The arraylists are chosen because they give flexibility in adding the new vertices (shrinkened loop has to be represented as a node). Similarly used arraylists to store the visiting status of the each vertex.
2. To find the minimum incoming weight of a vertex iterator is taken on a linked list and the arraylist is iterated to find the minimum among them and this is subtracted from all the incoming edges. This is done for all the vertices.
3. While walking backwards on a zero weighted edges we come across the zero weighted loops and to shrink these loops we have to save all the incoming and outgoing edges of the vertices in the loop so two lists were taken to save the edges which will be effected by shrinking. One for outgoing and incoming edges to outside of these loop and the other list to store the loop edges itself.
4. After shrinking the loop to a single vertex (Vertex with node number one greater than the current no of vertices) the loop info is push on to a stack (stored globally) so that we can retrieve this information while expanding the node by popping out elements from the stack.
5. Once the shrinking is done, so the edges linked to the loop and set as disabled (Edge class have setinAdjacency variable to do this). The edges were initially set to enable by default.
6. While expanding the loop from a particular vertex (vertex created by shrinking the loop) we pop the corresponding savings from the stack and enable those edges and disable the edges involved with the shrinked vertex.
7. For printing purpose the edges are stored in a hash map as Strings and these edges are printed sequentially sorted by their head when number of edges is less than 50.

Functions implemented :

1. DFS: The dfs is modified to implement a path that contains only zero length edges. A stack is used to traverse all the paths from a particular node. Once all the zero length paths from a particular node is traversed then the node is popped out.

Test results – The test cases are all tested with the given inputs and got correct output for all the cases but for the cases which the detailed output has to be produced there is some error in printing the path for large input cases , for the intial input case (1st one and the inputs with nearly 10 nodes correct output is produced).

References :

1. For DFS
   http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/Graph/Progs/dfs/Graph2.java