Queens College of CUNY
Department of Computer Science
Design and Analysis of Algorithms (CSCI 323)
June 2024

Assignment #9
"Minimum Spanning Tree Algorithms"
Due: June 24, 2024

Overview:

This assignment builds on the graph-related functions of a previous assignment, as well as the general infrastructure of several earlier assignments, to implement and study the empirical performance of several algorithms for the Minimum Spanning Tree (MST) problems, namely

- Prim's MST Algorithm w/ matrix
- Prim's MST Algorithm w/ table
- Kruskal's MST Algorithm w/ matrix
- Kruskal's MST Algorithm w/ table

Submissions:

Use the Google form to submit the following:

- Assignment09.py (source code)
- Assignment09.txt (console output)
- Assignment09-times.png (bar graph of timings)
- Assignment09-graph.png (graph w/ MST)

Tasks:

Follow the template and general guidelines for previous assignments. Wherever possible, import those assignments and invoke their functions rather than creating and maintaining copies of the same code.

[1] Define a function print_mst() that prints the MST, listing each edge used and its weight in a nice columnar format, and then the total weight of the MST on the bottom. It should also state the number of vertices and edges in the MST.

[2] Define a function prim_mst_matrix(graph) that finds the MST of the graph using Prim's MST algorithm.
See https://www.geeksforgeeks.org/prims-min_matriximum-spanning-tree-mst-greedy-algo-5/

[3] Define a function prim_mst_table(graph) that finds the MST of the graph using Prim's MST algorithm.
See https://www.geeksforgeeks.org/prims-algorithm-using-priority_queue-stl/

[4] Define a function kruskal_mst_matrix(graph) that finds the MST of the graph using Kruskal's MST algorithm
See https://www.geeksforgeeks.org/kruskals-algorithm-simple-implementation-for-adjacency-matrix/

[5] Define a function kruskal_mst_matrix(graph) that finds the MST of the graph using Kruskal's MST algorithm
See https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

[6] Use our general infrastructure to execute the various versions of these MST algorithms for different sizes; tabulate and plot their runtimes

[7] Define a function draw_mst(graph, mst) that superimposes the mst in one color over the graph in another color