University of Illinois at Urbana-Champaign
Department of Industrial and Enterprise Systems Engineering
IE 533 Big Graphs and Social Networks (Spring 2023)
Homework #2
Assigned 1/26 Thursday
Due 2/2 Thursday

Note: Prove if and only if statements in both directions

1. Show that a directed graph *G* is acyclic if and only if we can renumber its nodes so that its node-node adjacency matrix is a lower triangular matrix.

2. Let *M* denote the node-node adjacency matrix of a network *G*. Show that *G* is strongly connected if and only if the matrix $M + M^2 + M^3 + \ldots + M^n$ has no zero entry, for some integer *n*.

3. Consider an undirected network with an origin, a destination, and 100 additional nodes, with each pair of nodes connected by an arc. Show that the number of different paths from origin to destination is given by

$$100! + 100\,(99!) + \binom{100}{2}(98!) + \cdots + \binom{100}{98} 2! + 100 + 1.$$

4. Show that every tree forms an undirected bipartite graph. Does every undirected bipartite graph form a tree, prove or give a counter example?

Programming Question:

1. For this part of the homework, you will implement some of the algorithms learnt in class. You may use Python, C++, or Java to execute this (Prefer C++ as it will be helpful with CUDA later). Custom packages cannot be used to replace the algorithms tested. The dataset for this can be downloaded from here. This source contains an undirected graph with 112 nodes and 425 edges.
   a. Store the graph in adjacency matrix form.
   b. Write a program to answer the following questions: Is this graph connected? What is the median degree of the nodes?
   c. Using the adjacency matrix, write a program to compute the number of triangles. Note: A *triangle* is a connected subgraph of size three.

   To set up instance of minimum spanning tree, add edge costs using the following procedure. For an edge between nodes with indices *i* and *j,* let |*i*-*j*| be its edge cost. For example, the cost associated with edge (31,17) is 14. Using this modified graph:

   d. Implement the Kruskal's algorithm to find the minimum spanning tree.
   e. Implement the Prim's algorithm to find the minimum spanning tree.
   f. Report the performances for each algorithm (running time, optimal cost, and the optimal solution). Why is there a difference in the run time?
   g. Formulate the minimum spanning tree problem as a polynomial sized mixed integer linear program (MILP).
   e. For the given graph instance, solve the MILP formulation using a commercial solver of your choosing, and report the results (running time, optimal cost, and the optimal solution). How does this compare with part f?