Ehsan Gholami
Hugo Mailhot

ECS 289I
Assignment 1

Q1

A tree of order *n*, denoted by T_n, is a simple connected acyclic graph, with the following properties:

1. **T_n is connected.**
2. **T_n is acyclic.**
3. **T_n has exactly *n-1 edges.***

Prove that any *2*-combination of the properties listed above, is both necessary and sufficient for the third property to be true.

<u>Connected + acyclic → n-1 edges</u>
Let G be a connected acyclic graph with n nodes and m edges. We will show that its connectedness necessarily entails that m >= n-1, and that its acyclicity necessarily entails that m <= n-1. Taken together, those two properties are then sufficient to guarantee that m = n-1.

Assume that Graph G has n nodes and m edges. We construct the graph one edge at a time by the following:

We call the constructed graph at step k, G_k. G_0 is the empty graph. Pick the first node randomly from the nodes and put it in G_1. Since G is connected, for all k < n, at step k there exists a node in remaining set of nodes that has an edge to G_{k-1}. Choose that node and edge and add it to G_{k-1}. From the second step until the n-th step, we are adding one node and one edge exactly at each step. At step n, all the n nodes are included in G_n which includes n-1 edges. The minimum number of edges in a connected graph is thus n-1.

Remember now that G is acyclic. We now show that no more edges could be added while retaining acyclicity. Pick any two nodes at random, u and w. Since u and w are already in G_k and that G_k is connected, there exist a path $P(u -> w)$ in G_k from u to w, and adding an edge (u, w) will create a second path $P(u -> w)$, thus introducing a cycle in G_{k+1}. This contradicts our acyclicity assumption. Therefore, G cannot have more than n-1 edges.

Connectedness and acyclicity are thus sufficient to guarantee n-1 edges.

<u>Connected + n-1 edges → acyclic</u>

Ehsan Gholami
Hugo Mailhot

Suppose this proposition is false. Then this means there exists a connected *cyclic* graph G with $n-1$ edges, where n is the number of vertices. Let C be a subset of G that forms a simple cycle of m vertices. Then C has m edges. There is $n-m$ vertices in $G \setminus C$, and since G is connected, each of the $n-m$ vertices in $G \setminus C$ needs at least one edge to be connected to a component that includes C (refer to part one for proof). Thus there are at least $m+(n-m)=n$ edges in G, which contradicts our assumption of there being $n-1$ edges. It is therefore impossible for a connected graph with $n-1$ edges to be cyclic.

<u>Acyclic + n-1 edges → connected</u>
Proof by contradiction. Assume graph G is acyclic and has n-1 edges but it is not connected. Assume G has $k>1$ connected components, each with m_i ($i=1..k$) nodes. Thus, $\sum_{i=1}^{k} m_i = n$.
Since the whole graph is acyclic, each component is acyclic too. Therefore, according to part 1, each component is connected and acyclic, and will have $m_i - 1$ edges. The total number of edges for G will be:

$$\sum_{i=1}^{k}(m_i-1) = \sum_{i=1}^{k} m_i - k = n-k$$

On the other hand we know G has $n-1$ edges. Therefore, $k=1$, which means G has 1 connected component and G is connected.

Q2
A sequence of *n* positive integers d*1* ≤ d*2* ≤ … ≤ d*n-1* ≤ d*n* constitutes a *graphic* sequence, if there exists a graph *G(V, E)* with {d*1*, d*2*, … , d*n-1*, d*n*} as vertex degrees. Prove that *G(V, E)* is a tree, if and only if, d*1* + d*2* +… + d*n-1* + d*n* = *2* (*n – 1*).

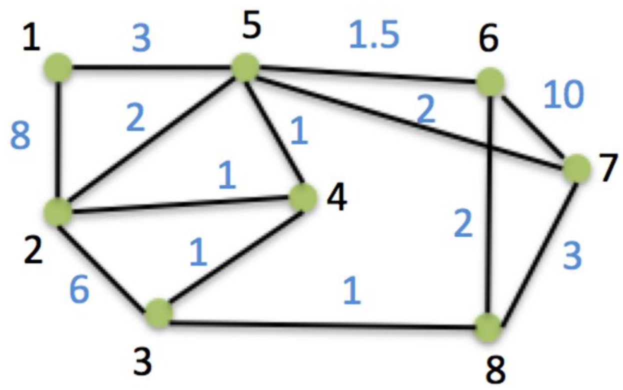
$G(V,E)$ *is a tree* $\rightarrow d_1 + \ldots + d_n = 2(n-1)$:
If $G(V,E)$ is a tree, according to first problem we know it has $n-1$ edges. Each edge contributes twice in total degree count as it has two nodes attached to its ends. Therefore, total degrees of the nodes would be $2(n-1)$.

$d_1 + \ldots + d_n = 2(n-1) \rightarrow G(V,E)$ *is a tree* :
Since the sum of degrees add up to 2(n-1), $G(V,E)$ has n-1 edges. By assumption $G(V,E)$ is connected. Therefore, by problem 1, $G(V,E)$ is a tree.

Q3

Ehsan Gholami
Hugo Mailhot

Consider the following graph *G(V,E)*:



1. **Write a program to compute the shortest path between any two vertices in *G*.**

   See program *hw1_SP_dijkstra.py*

2. **Rank the vertices of *G* by degree, geodesic centrality (closeness), and shortest path betweenness centrality. Compare and contrast these rank orders.**

Degree		**Shortest path betweenness centrality**		**Geodesic centrality**	
Node	Value	**Node**	Value	**Node**	Value
5	3	**5**	11.5	**5**	0.069
4	3	**4**	8.5	**4**	0.069
3	2.17	**3**	4	**3**	0.056
8	1.83	**8**	1.5	**8**	0.05
2	1.79	**2**	0	**2**	0.049
6	1.27	**6**	0	**6**	0.049
7	0.93	**7**	0	**7**	0.041
1	0.46	**1**	0	**1**	0.031

These centrality measures were computed using R's igraph library. We used the reciprocal of edge weights to compute degree centrality.

First of all, as we made clear in the above table, all rankings can be made equivalent by ordering the ties arbitrarily. So, for this particular graph, there is no necessary contradiction between these rankings.

The rankings also agree on nodes 4 and 5 being much more central than other nodes, while node 1 is adequately identified as not central. Shortest path centrality is the only one to break the tie between 4 and 5.

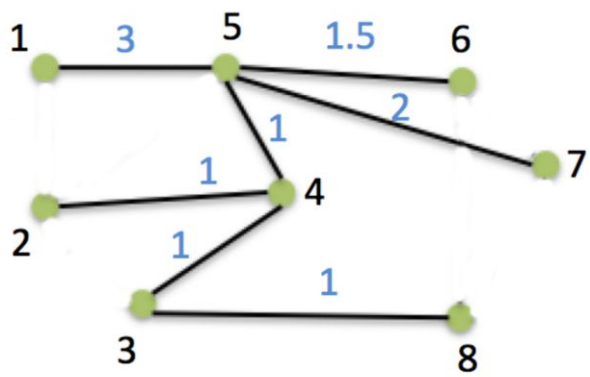
As expected, shortest path centrality is unfair to nodes that don't lie in shortest paths. It is also *equally* unfair, in that half of the nodes get the same low score of zero. In that regard, we can say that shortest path centrality is not the best choice to attain some total order of node centrality, but is rather best used to identify only the most central. To break such ties, we could use closeness centrality or degree centrality.

3. **Write a program to compute the Minimum Spanning Tree of *G*.**

   See program *hw1_MST_prim.py*

   Result: (5, 1), (4, 5), (2, 4), (3, 4), (8, 3), (6, 5), (7, 5)

   Cost: 10.5



Can the algorithm be changed to compute minimum spanning forests?

Yes. To compute MSF with n trees in it, compute MST, then remove from it the n-1 edges with highest weight. By doing so you disconnect the MST into n trees, minimizing the cost by as much as possible, which gives you a MSF.