6.1 Directed Acyclic Graphs

A graph with no cycles is **acyclic**. **Directed acyclic graphs**, or **DAGs** are acyclic directed graphs where vertices can be ordered in such at way that no vertex has an edge that points to a vertex earlier in the order. This also implies that we can permute the adjacency matrix in a way that it has only zeros on and below the diagonal. This is a *strictly* **upper triangular** matrix. Arranging vertices in such a way is referred to in computer science as **topological sorting**.

6.2 Trees

A **forest** is an undirected acyclic graph. A **tree** is a connected undirected acyclic graph. If the underlying graph of a DAG is a tree, then the graph is a **polytree**. A **leaf** is a vertex of degree one. Every tree with at least two vertices has at least two leaves. A **spanning subgraph** of G is a subgraph that contains all vertices. A **spanning tree** is a spanning subgraph that is a tree. All graphs contain at least one spanning tree.

Necessary properties of a tree T:

1. T is (minimally) connected
2. T is (maximally) acyclic
3. T has $n-1$ edges
4. T has a single u, v-path $\forall u, v \in V(T)$
5. Any edge in T is a cut edge; any non-leaf vertex is a cut vertex
6. Adding any edge to a tree (without adding vertices) creates a cycle

Which of these properties are also sufficient?

Generally, because any possible tree can be constructed by iteratively adding vertices and edges to the simplest possible tree (single vertex/single edge), we can often use weak (instead of strong) induction in proofs on trees.

We can prove using weak induction that every tree is bipartite.

We can also prove using strong induction on the number edges m of a tree T: Between any two $u, v \in V(T)$, there exists only a single unique u, v-path. Note that the approach in this proof is nearly identical to what we would do with weak induction.

Note: In prior classes (especially the CS students), you might have used *structural induction* in the context of proofs on trees. The difference with structural induction is that there does not need to be an explicit "countable" variable on which induction is performed, though the recursive idea behind the proof technique is similar to strong induction. We won't be covering structural induction in this course.

6.3 Distances

The **distance** between a u and v in G written as $d(u, v)$ is the length of the shortest u, v-path. Remember that the **diameter** of a graph is $max_{u,v \in V(G)} d(u, v)$, or the maximum $d(u, v)$ among all possible u, v pairs. The **eccentricity** of a vertex $\epsilon(u)$ is $max_{v \in V(G)} d(u, v)$, or maximum u, v path from that u. The **radius** of a graph is the minimum eccentricity of any vertex, or $min_{v \in V(G)} d(u, v)$. The **center** of a graph is the subgraph induced by the vertices of minimum eccentricity.