Tree

1.1 Properties of Trees:

Definition: A graph G = (V, E) is called a tree if G is connected and acyclic.

The following theorem captures many important facts about trees.

Theorem: (Characterizations of trees)

Let G = (V, E) be an undirected graph having n vertices and m edges. The following statements are equivalent.

1. G is a tree.
2. There is a unique path between any two vertices in G.
3. G is connected but G-e is disconnected for every edge e of G.
4. G is connected, and m=n-1.
5. G is acyclic, and m=n-1.
6. G is acyclic but G + xy is cyclic for every x, $y \in V$ with $xy \notin E$.

Proof: (1)$\Rightarrow$(2): Since every tree is connected, there is at least one path between any two vertices in G. Hence, to show that there is a unique path between any two vertices in G, we have to show that there is at most one path between any two vertices in G. We prove this by contradiction. So, assume that there are at least two paths between some pair of vertices, say between x and y. Let P_1 and P_2 be two distinct paths from x to y. By lemma 2.1, $P_1 \cup P_2$ contains a cycle. So, G contains a cycle. This contradicts the fact that G is a tree. Hence, there is a unique path between any two vertices in G.

(2)$\Rightarrow$(3): Since, any two vertices in G are connected by a unique path, G is connected. Let xy be any edge in E. Then, P=xy is a path from x to y. So, it must be a unique path from x to y. If we remove xy from G, then there is no path from x to y. Hence, G-xy is disconnected. Since, xy is a arbitrary edge of G, G-e is disconnected for every edge e of G. Hence, G is connected but G-e is disconnected for every edge e of G.

(3)$\Rightarrow$(4): By assumption, G is connected. So we need only to show that m=n-1. We prove this by induction. A connected graph with n=1 or n=2 vertices has n-1 edges. Assume that every graph with fewer than n vertices satisfying (3) also satisfy (4). Suppose that G has $n \geq 3$ vertices and G satisfies (3), i.e. G is connected but G-e is disconnected for every edge e of G. Let e=xy be any edge of G. Now, G-e is disconnected. Now, by lemma 3, G-e has exactly two connected components. Let G_1 and G_2 be the connected components of G. Let n_i. and m_i, $1 \le i \le 2$, be the number of vertices and edges in G_i, $1 \le i \le 2$. Now, each component satisfies (3), or else G would not satisfy (3). Since, $n_i < n$, i=1,2, by induction hypothesis, $m_i = n_i - 1$, $1 \le i \le 2$. So, $m = m_1 + m_2 + 1 = (n_1-1) + (n_2-1) + 1 = n-1$. So, by induction principle, G has exactly n-1 edges.

(4)$\Rightarrow$(5): We have to show that every connected graph G with n vertices and n-1 edges are acyclic. We prove this by induction. For n=1, 2 and 3, it can be easily checked that all connected graph with n vertices and n-1 edges are acyclic. Assume that every connected graph with fewer than n vertices satisfying (4) is acyclic. Let G be a connected graph having n vertices and n-1 edges. Since, G is connected and has n-1 edges, G has a vertex of degree 1. Let x be a vertex of degree 1 in G. Let G′=G-x. Now, G′ is connected and has n-1 vertices and n-2 edges. So, by induction hypothesis, G′ is acyclic. Since, x is a degree 1 vertex, x can not be in any cycle of G. Since G′=G-x is acyclic, G must be acyclic. So, by induction, every connected graph with n vertices and n-1 edges is acyclic.

(5)$\Rightarrow$(6): Suppose that G is acyclic and that m=n-1. Let G_i, $1 \le i \le k$ be the connected components of G. Since G is acyclic, G_i is acyclic for $1 \le i \le k$. Hence, each G_i, $1 \le i \le k$ is a tree. Let n_i and m_i, $1 \le i \le k$, be the number of vertices and edges in G_i, $1 \le i \le k$, respectively. Since (1) implies (5), $m = \sum_{i=1}^{k} m_i = \sum_{i=1}^{k} (n_i - 1)$ =n-k. So, k=1. So, G must be a tree. Since (1) implies (2), any two vertices in G are connected by a unique path. Thus, adding any edge to G creates a cycle.

(6) $\Rightarrow$(1): Suppose that G is acyclic but G +xy is cyclic for every x,y in V with xy $\notin$ E. We must show that G is connected. Let u and v be arbitrary vertices in G. If u and v are not already adjacent, adding the edge uv creates a cycle in which all edges but uv belong to G. Thus, there is a path from u to v and since u and v were chosen arbitrarily, G is connected. □

Exercises 1.1

1. If the maximum degree in a tree T is k, then prove that T has at least k pendant vertices (vertices of degree 1). Is the converse true?
2. Let T_1 and T_2 be two spanning trees of a connected graph G. If edge e is in T_1 but not in T_2, prove that there exists another edge f in T_2 but not in T_1 such that $(T_1\text{-}e) \cup f$ and $(T_2\text{-}f) \cup e$ are also spanning trees of G.
3. Prove that in a tree every vertex of degree greater than one is a cut vertex.
4. Prove that a pendant edge in a connected graph G is contained in every spanning tree of G.
5. Prove that an edge e of a connected graph G is a cut edge if and only if e belongs to every spanning tree.
6. Let T be a tree of order m, and let G be a graph with $\delta(G) = m-1$.Then prove that T is isomorphic to some sub graph of G.

7. Suppose T is a tree of order n that contains only vertices of degree 1 and 3. Prove that T contains (n-2)/2 vertices of degree 3.
8. Prove or disprove: if d_1, d_2,…d_n is the degree sequence of a tree, then 1, d_1+1, d_2, d_3,… .d_n is the degree sequence of a tree.
9. Let G be a connected weighted graph whose edges have distinct weights. Show that G has a unique minimum spanning tree.
10. Let T be a tree of order n and size m having n_i vertices of degree i (i=1, 2…).show that n_1= n_3+2n_4+3n_5+4n_6+….+2.
11. Prove or disprove: if n_i denotes the number of vertices of degree i in a tree T, then Σ i n_i depend only on the number of vertices in T.
12. Let T be an n vertex tree having one vertex of each degree i , 2 $\leq$ i $\leq$ k; the remaining n-k+1 vertices are leaves. Determine n in terms of k.
13. Draw a weighted connected graph G on 11 vertices having 10 different MSTs.
14. Let e be a minimum cost edge of a weighted connected graph G. Show that e belongs to some MST of G.
15. If e be the only minimum cost edge of G, then e belongs to every MST of G.
16. Describe five applications of MST.
17. Design algorithms for a Tree for each of the following:
    1. To find a maximum independent set.
    2. To 2-color all the vertices of G.
    3. To find a path from x to y.
18. Suppose n $\geq$2 and d_1, d_2, …,d_n,dn+1 are n+1 positive integers such that their sum equals 2n. Use the pigeon principle to prove that there exists an index i such that d_i= 1 and there is an index j such that d_j > 1.
19. Use Q18. and Mathematical induction to show that in n is an integer $\geq$ 2 and d_1, d_2, …,d_n are positive integers such that $\sum_{i=1}^{n} d_i$ =2n-2, then there is a tree T_n with n vertices whose degrees are d_1, d_2, …, and d_n.
20. Characterize all connected graphs with same number of vertices and edges.

1.2: Spanning Tree:

Definition 2: A subgraph T= (V_1, E_1) of a graph G = (V,E) is a spanning tree if
(i) T is a tree, and
(ii) V_1=V.

Theorem 3.1: A graph admits a spanning tree if and only if G is connected.

Proof: Necessity:
Suppose G admits a spanning tree, say T. We will show that G is connected. Let u, v be any two arbitrary vertices of G. Since, T is a spanning subgraph of G, u and v are vertices of T as well. Since, T is connected, there is a path P(u,v) from u to v in T. As T is a subgraph of G, P(u,v) is also a path in G. Since, u and v are arbitrary vertices of G, there is a path between any two vertices of G. Hence G is connected.

Sufficiency: Let G be a connected graph with n vertices and m edges. We construct a spanning tree T of G. Let k=m-n+1. Define G_i, $0\le i \le k$, recursively, as follows:

$$G_i = \begin{cases} G \text{ if } i = 0, \\ G_{i-1} - e_i \text{ where } e_i \text{ is an edge in some cycle of } G_{i-1} \text{ if } 1 \le i \le k. \end{cases}$$

Since, G_i has exactly n-1+k-i edges, G_i is cyclic for each i, $0\le i \le k-1$. So, each G_i, $0\le i \le k-1$, has a cycle. If G_{i-1} is connected, then G_i is also connected, as e_i belongs to some cycle of G_{i-1}, $0\le i \le k-1$. Hence, Gk is connected and has exactly n-1 edges. So, Gk is a tree. Let T=Gk. Now T is a spanning tree of G. □

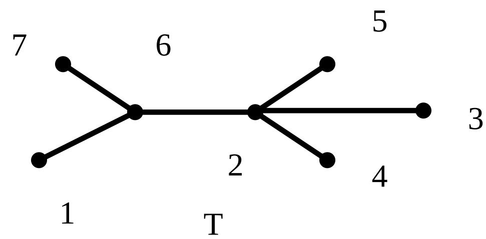
Let $\tau(G)$ denotes the number of distinct spanning trees of a graph. So, by theorem 3.1, $\tau(G) \ge 1$ for a connected graph. The following graph has exactly three spanning trees.

The following theorem gives the number of distinct spanning trees of a complete graph.
Theorem 3.2: $\tau(K_n) = n^{n-2}$ for all $n \ge 1$.

Proof: Since each of K1 and K2 has exactly one spanning tree $\tau(K_n) = n^{n-2}$ for n=1 and 2. So assume that $n \ge 3$. Assume that V (K_n) = {1, 2…,n}. Let X be the set of all spanning trees of K_n and Y be the set of all sequences $a_1,a_2,\ldots,a_{n-2}$ of length n-2 such that $a_i \in \{1,2,\ldots,n\}$. Note that Y has n^{n-2} sequences as each a_i , $1\le i \le n-2$ can be selected in n different ways. So, to show that there are n^{n-2} spanning trees of K_n, it is enough to produce a function f: $X \to Y$ which is a bijection.

Let T be any spanning tree of Kn. Define f(T)= $a_1,a_2,\ldots,a_{n-2}$,a unique sequence of length n-2 such that $a_i \in \{1,2,\ldots,n\}$ for each i, $1\le i \le n-2$, in the following way. Among all the vertices of degree one, let s_1 be the vertex such s_1

as an integer is minimum. Let t_1 be the vertex adjacent to s_1 in T. Assign t_1 to s_1. Then, delete the vertex s1 from T. Next, among all the vertices of degree one in T-$\{s_1\}$, let s_2 be the vertex such s_2 as an integer is minimum. Let t_2 be the vertex adjacent to s_2 in T-s_1. Assign t_2 to a_2. Then, delete the vertex s_2 from T-$\{s_1\}$. Repeat this process until a_{n-2} has been defined and a tree with just two vertices remains; the tree T in figure 1, for instance, gives rise to the sequence 6,2,2,2,6.



T

To show that f is a bijection, we have to prove that (i) no sequence is produced by two different spanning trees of K_n and (ii) every sequence of Y is produced by some spanning tree of Kn. We shall achieve both (i) and (ii) by showing that f has an inverse, i.e. we can construct a spanning tree of K_n from a sequence $a_1,a_2,\ldots,a_{n-2}$ of Y by reversing the process described above of obtaining a sequence of length n-2 from a spanning tree of K_n.

Let T be any spanning tree of K_n, and let f(T)= $a_1,a_2,\ldots,a_{n-2}$. Then, d(k), the degree of vertex k in T, is equal to the number of times k appears in the sequence $a_1,a_2,\ldots,a_{n-2}$, plus 1. This follows from the observation that when each, but the last, of the edges incident on k is deleted, k appears in the sequence; the last edge may never be deleted, if k is one of the two vertices remaining in the tree, or if it is deleted, k is now the removed leaf, and the adjacent vertex, not k, is included later in the sequence. Thus, if k appears in the sequence then the degree of k in T must be as stated.

For example, if f(T) =6,2,2,2,6 is the sequence obtained from a tree T, then d(6)=3, d(2) = 4, while d(1)=d(3)=d(4)=d(5)=1.

So, let f(T)= $a_1,a_2,\ldots,a_{n-2}$. We reconstruct T as follows.

Let s_1 be the vertex such that s1 is the least integer in {1, 2,…,n} that does not appear in the sequence $a_1,a_2,\ldots,a_{n-2}$. Join s1 to a1. Then, let s_2 be the vertex such that s2 is the least integer in {1, 2,…, n}-$\{s_1\}$ that does not appear in the sequence $a_2,\ldots,a_{n-2}$. Join s_2 to a_2. Follow this procedure until s_{n-2} is obtained from the sequence a_{n-2}. Join s_{n-2} to a_{n-2}. The tree T is obtained by adding the edge joining the two remaining vertices of N-$\{s_1, s_2,\ldots,s_{n-2}\}$. For example, the tree T for which f(T)=6,2,2,2,6 is given in figure 1.

Thus, we have now established the required one-one correspondence. Hence, the number of distinct spanning tree of K_n is n^{n-2}. □

1.2 Minimum Spanning Tree;

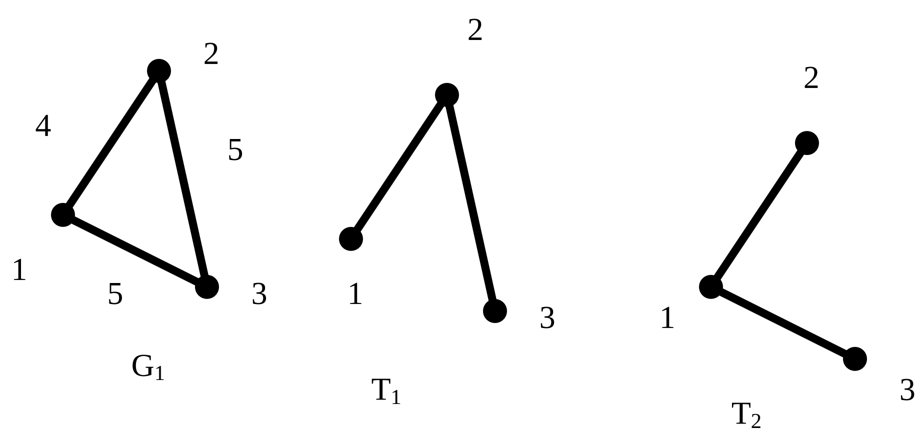
Let G=(V,E) be a connected weighted graph and C be the cost matrix of G. Let T=(V,E) be a spanning tree of G. The cost of T, denoted C(T), is defined as follows:

Definition 3: The cost of a spanning tree T=(V, E′) of a weighted graph G=(V,E) with cost matrix C is defined by

$$C(T)=\sum_{e\in E'} C(e) \ .$$

Definition 4: A spanning tree T of a weighted connected graph is called a minimum spanning tree if $C(T) \le C(T')$ for any other spanning tree T′ of G.

Note that there many be more than one minimum spanning tree of a graph. For example, the graph G in the following figure has exactly two minimum spanning trees.



A graph G having two minimum spanning trees.

1.4 Minimum Spanning Tree Algorithms

In this section, we will discuss two popular algorithms to construct a minimum spanning tree of a weighted connected graph.

1.4.1 Kruskal's Algorithm

As we have seen in the previous section that an acyclic graph with n vertices and n-1 edges is a tree. The first algorithm, known as Kruskal's Algorithm, uses this fact to construct an MST of a connected weighted graph G.

We first describe the algorithm informally. First, the algorithm arranges the edges of the graph G in the non-decreasing order of their costs. It starts with the graph T=(V,E′), where E′ = ϕ initially. It then examines each edge for inclusion into the T. If the current edge e under examination does not form a cycle with the so far selected edges, then the edge e is included in T. If e forms a cycle with the so far selected edges, then e is rejected. After the decision of selecting or rejecting the current edge e, the next edge in the list becomes current edge. The algorithm terminates once n-1 edges have been selected or there is no edge left for consideration. The algorithm, thus, maintains acyclicity at each stage of inclusion of edge to T. If the algorithm is successful in adding n-1 edges to T, then T becomes a spanning tree of G. Since the edges are examined for inclusion in the non-decreasing order of their costs, T turns out to be an MST.

We, next describe the Kruskal's Algorithm formally.

Kruskal's Algorithm

Input: A connected Graph G=(V,E) and the cost matrix C of G.
Output: A Minimum spanning tree T=(V,E′) of G.

Method:

Step 1: Sort the edges of G in the non-decreasing order of their costs. Let the sorted list edges be $e_1, e_2, \ldots, e_m$.

Step 2: T=(V, E′), where E′=ϕ. i=1; count=0;

```
while ( count < n-1 and i < m)
{
 if ( T=(V, E′ ∪ { e_i }) is acyclic)
     { E′ = E′ ∪ { e_i }; count= count+1;}
 i=i+1;
}
```

Next, we prove that the graph T=(V,E′) obtained by Kruskal's algorithm is in fact a minimum spanning tree.

Theorem 1.4.1: Kruskal's Algorithm produces a minimum spanning tree in a connected weighted graph.

Proof: Let G = (V,E) be a connected weighted graph and let T=(V,E′) be the subgraph produced by Kruskal's algorithm for G. Each edge is added to T by Kruskal's algorithm if it does not form a cycle with the already added edges. Hence, T must be acyclic. Next we show that T is connected. If possible T is disconnected and let G_1= (V_1, E_1) be a connected component of T. Let V_2=V-V_1. Let $f_1,f_2,\ldots,f_k$ be the edges of T such that cost(f_i) $\leq$cost (f_{i+1}). Since T is disconnected, k < n-1. Let E_2=(xy∈E | x ∈V_1 and y ∈ V_2}. Since, G is connected, E2 is non empty. Let e ∈ E_2 be the least cost edge among all edges of E_2. Let $f_1, f_2,\ldots,f_i$ be the edges that have already been selected by Kruskal's Algorithm when the edge e was examined. Since T +e is acyclic, e does not for a cycle with the edges $f_1,f_2,\ldots,f_i$. So, Kruskal's algorithm would have selected the edge e after selecting the edge f_i. This is a contradiction to the fact that T is the graph produced by Kruskal's algorithm. Hence, T must be connected. Hence, T is a spanning tree.

Next, we show that T has least cost among all spanning trees of G. We prove this by contradiction. Suppose, to the contrary, that T is not a minimum spanning tree. Let E(T)={ $f_1,f_2,\ldots,f_{n-1}$ } such that w(f_i) $\leq$ w (f_{i+1}), 1≤ i ≤ n-2. Note that G may have more than one minimum spanning tree. Let T_1 be a minimum spanning tree of G having maximum number of edges in common with T. Let i be the smallest index, 1≤ i ≤ n-1 such that f_i is not an edge in T. Such an index exists since T and T_1 are two distinct trees of G. Let T_2= T_1+f_i. Now, T_2 has a unique cycle C containing f_i. Note that C contains at least one edge e_0 such that e_0 is not an edge of T. Let T_3=T_2-e_0. Now T_3 is a spanning tree of G and W(T_3)= W(T_1)+W(f_i)-W(e_0). Since, W(T_1) ≤W(T_3), we have w(e_0) ≤w(f_i). By Kruskal's algorithm, fi is an edge of minimum cost such that G[{$f_1, f_2,\ldots,f_{i-1}$} ∪ {f_i}] is acyclic. However, G[{$f_1, f_2,\ldots,f_{i-1}$} ∪ {e_0}] is a subgraph of T_1, and hence acyclic. So, w(e_0)=w(f_i), otherwise Kruskal's algorithm would have chosen e_0 instead of f_i. Thus, W(T_3)=W(T_1). Hence, T_3 is a minimum spanning tree. But, T_3 has more edges in common with T than T_1 has with T. This contradicts the fact that T_1 has maximum number of edges in common with T than any other spanning tree of G. Hence, T must a minimum cost spanning tree. □

We, next, consider another popular algorithm for finding a minimum spanning tree in a connected weighted graph.

1.4.2 Prim's Algorithm

As we have seen above, Kruskal's algorithm starts with the vertex set V and empty edge set and keeps on adding edges maintaining acyclicity throughout. Once n-1 edges are added, it becomes a tree because of the fact that an acyclic graph with n vertices and n-1 edges is a tree. Prim's algorithm adopts a different strategy. It uses the fact that a connected graph with n vertices and n-1 edges is a tree. It starts with vertex set V′= {v} where v is any arbitrary vertex of G and E′, where E′ = ϕ. It then selects a least cost edge e= xy with x ∈ V′ and y ∈ V- V′ from V′ to V- V′ and updates E′= E′∪{e} and V′= V′ ∪{y}. It stops when V′=V. Thus, it maintains throughout that G′=(V′, E′) is connected. Once V′=V, G′ becomes a spanning tree of G. We next describe the algorithm formally.

Prim's Algorithm

Input: A connected Graph $G=(V,E)$ and the cost matrix C of G.
Output: A Minimum spanning tree $T=(V,E')$ of G.

Method:

{

Step 1: Let u be any arbitrary vertex of G.

$T=(V',E')$, where $V' = \{u\}$ and $E' = \phi$.

Step 2: while ($V' \neq V$)

{

Choose a least cost edge from V' to $V-V'$.

Let $e=xy$ be a least cost edge such that $x \in V'$ and $y \in V-V'$.

$V'= V' \cup \{x\}$;

$E' = E' \cup \{ e \}$;

}

}

Next, we show that Prim's algorithm produces a minimum spanning tree of a connected weighted graph.

Theorem 2.3.2: Prim's Algorithm produces a minimum spanning tree in a connected weighted graph.

Proof: Let G be a connected weighted graph and let T be the subgraph produced by Prim's algorithm. Since, G is connected, T is a spanning tree of G. Next, we show that T is a minimum spanning tree of G. We prove this by method contradiction. Suppose, to the contrary, that T is not a minimum spanning tree of G. Let $E(T)=\{ f_1,f_2,\ldots,f_{n-1} \}$ such that $w(f_i) \leq w(f_{i+1})$, $1 \leq i \leq n-2$. Note that G may have more than one minimum spanning tree. Let T_1 be a minimum spanning tree of G having maximum number of edges in common with T. Let i be the smallest index, $1 \leq i \leq n-1$ such that f_i is not an edge in T. Such an index exists since T and T_1 are two distinct trees of G. For $i=1$, let $U=\{u\}$, where u is the first vertex added to V' by the Prim's algorithm. If $i \geq 2$, then let U be the vertex set of the subgraph induced by the edges $f_1,f_2,\ldots,f_{i-1}$. Now, f_i joins a vertex of U to a vertex of V-U. Let $T_2= T_1+f_i$. Now, T_2 has a unique cycle C containing f_i. The cycle C contains an edge e_0 that joins a vertex of U to a vertex of V-U. Let $T_3= T_1+f_i - e_0$. Then, T_3 is a spanning tree of G. Since f_i and e_0 are both edges from U to V-U and f_i is selected by Prim's algorithm, $w(f_i) \leq w(e_0)$. Therefore, $w(T_3) \leq w(T_1)$. Since, T_1 is a minimum spanning tree, T_3 is also a minimum spanning tree of G. But, T_3 has more edges in common with T than T_1 has with T. This contradicts the choice of T_1. Hence, T must be a minimum spanning Tree.□

Exercise

1. Let G be a connected weighted graph whose edges have distinct weights. Show that both Prims' algorithm as well as Kruskal's algorithm produces the same tree.
2. Show, for every integer $n \geq 2$, there a connected weighted graph having exactly n minimum spanning trees.