Discrete Structures

Trees

Trees

Definition: *A tree* is an undirected connected graph with no circuits.

Theorem: There are at least two vertices in a tree T with degree 1.

Proof: Consider the longest path in tree T with endpoints u and v. If the degree of u and v is greater than 1, we find either a longer path or a circuit. Both contradicts with T being a tree.

Theorem: Any tree T with n vertices has $n-1$ edges.

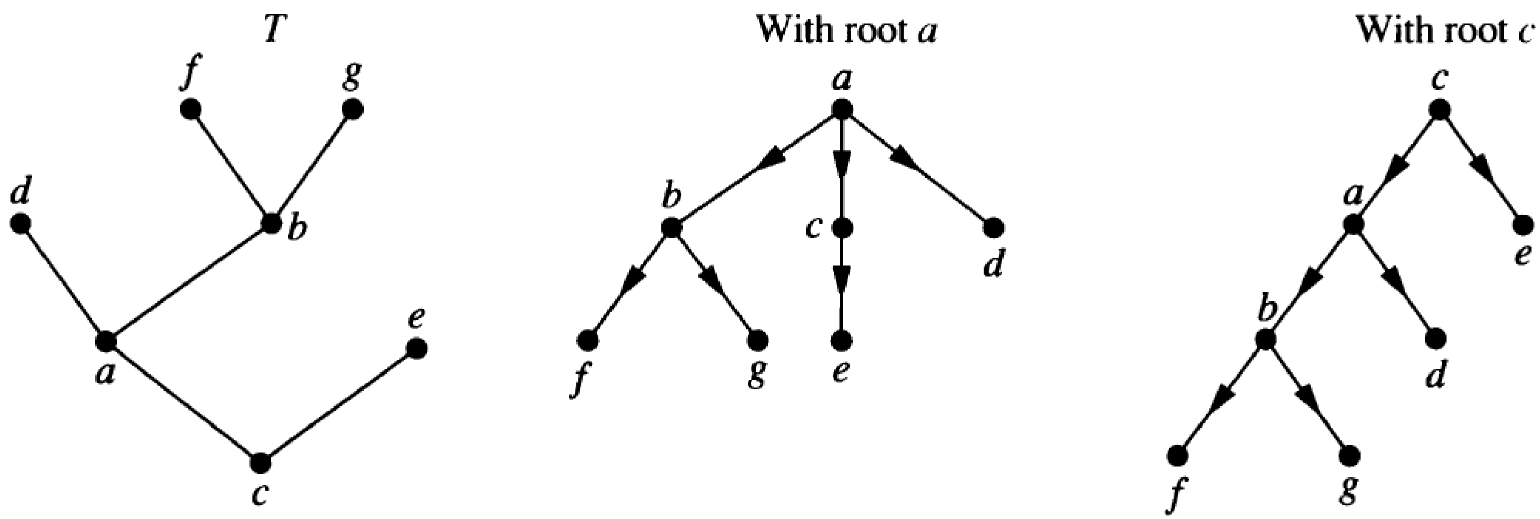
Proof: The proof is based on induction on n. T has a vertex u with degree 1. $T-u$ is a tree as well and based on the inductive hypotheses has $n-2$ edges. Then, T has $n-1$ edges.

Theorem: There is a unique path between any two vertices of a tree.

Rooted Trees

Definition: A rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

Note: We usually draw a rooted tree with its root at the top of the graph. The arrow indicating the directions of the edges in a rooted tree can be omitted, because the choice of root determines the direction of the edges.



Rooted Trees

Parent and Child: if there is a directed edge from u to v, then u is called the parent of v, and v is called a child of u.

Sibling: Vertices with the same parent are called siblings.

Leaf: A vertex without any child.

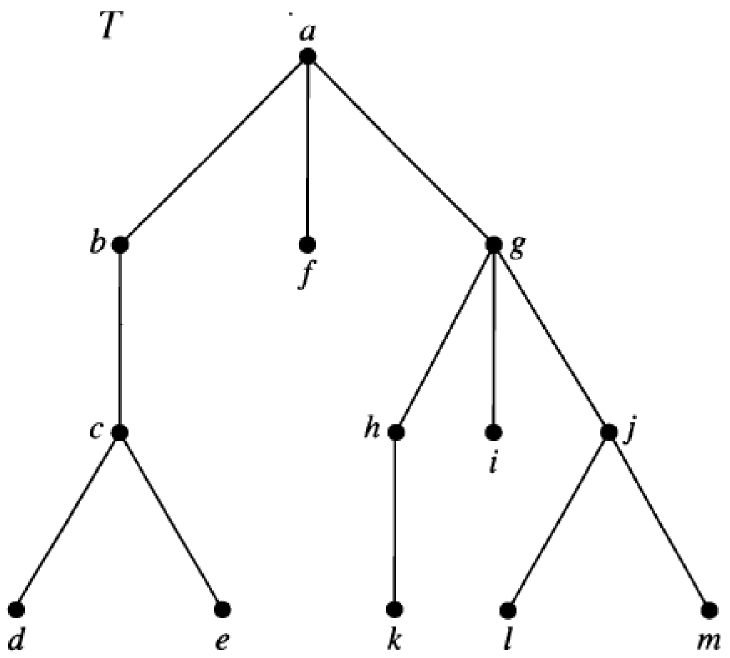
Ancestors: The ancestors of a vertex other than the root are vertices in the path from the root to the vertex excluding the vertex itself.

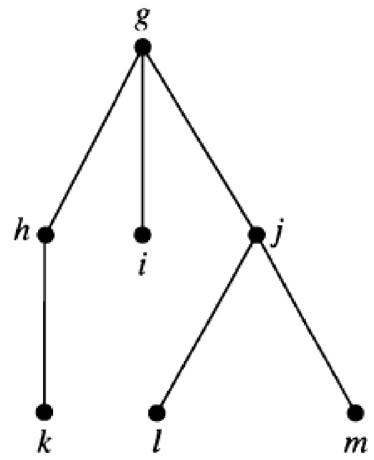
Descendants: The descendants of a vertex u are those vertices that have u as an ancestor.

Internal nodes: vertices with at least one child

Rooted Trees

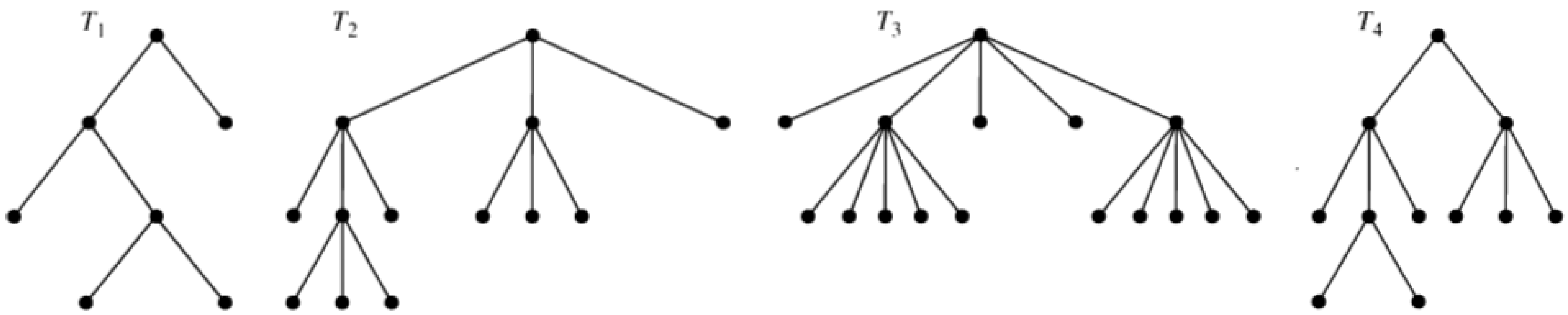
Subtree rooted at u: The tree including u and all its descendants





Rooted Trees

m-array tree: A rooted tree is called m-array tree if each internal node has at most m children. It is called a full m-array tree if each internal node has exactly m children. A 2-array tree is called a binary tree.



Theorem: A full m-array tree with i internal vertices contains $n = mi + 1$ vertices.

Proof: Except the root, each vertex is a child of another vertex. Since in total we have mi children, the tree contains $n = mi + 1$ vertices.

Rooted Trees

Ordered rooted tree: A rooted tree where the children of each internal node are ordered. For binary tree the children are called **left child** and **right child**. And the subtrees rooted at left child and right child are called **left subtree** and **right subtree** respectively.

Rooted Trees

The depth and height of a node u: The distance of the root to u is called the depth of u. The distance of u to the farthest leaf in the subtree rooted at u is called the height of u.

The height of a tree: The height of the root of the tree

A balanced rooted tree of height h: Any leaf is at depth h or $h-1$

Theorem: Any m-array tree of height h has at most m^h.

Proof: The proof can be simply done using induction. Just look at the subtrees rooted at children of the root. Each has height $h-1$ and then m^{h-1} leaves.

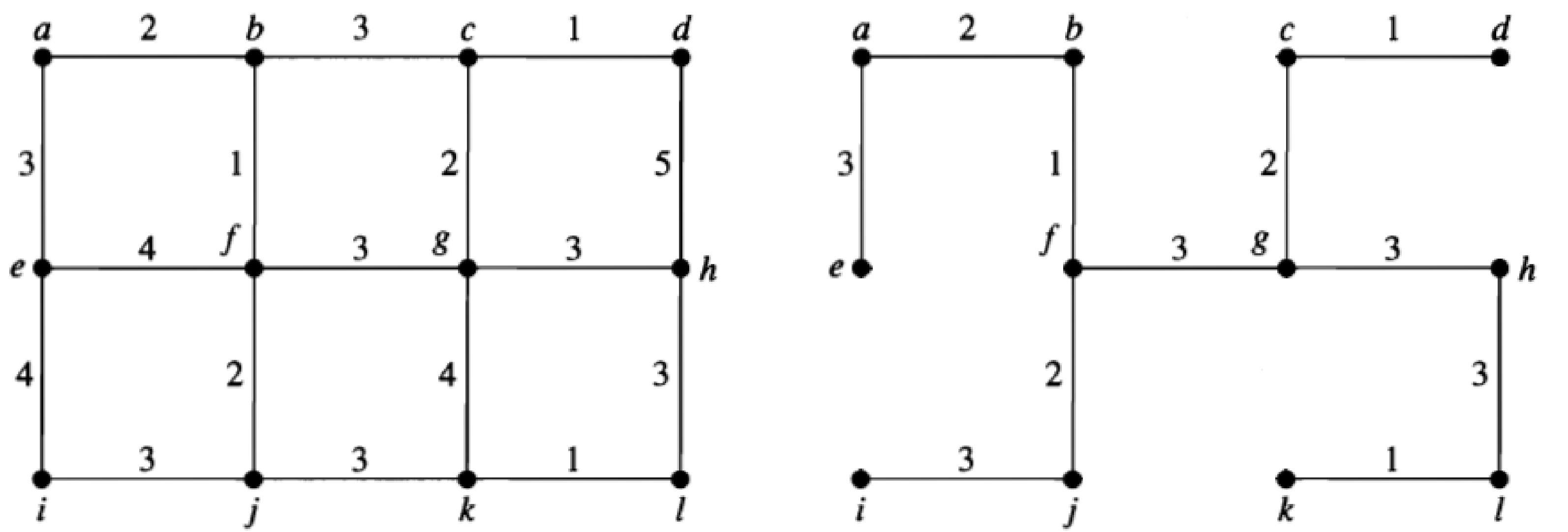
Corollary: If a m-array tree has l leaves, then the height of the tree is at least $\lceil \log_m l \rceil$.

Minimum Spanning Tree (MST)

A weighted undirected graph: A undirected graph $G(V,T)$ where each edge e is associated with a weight $w(e)$

A spanning tree: A subgraph $T(V',E')$ of a connected undirected $G(V,T)$ where $V'=V$ and T is a tree

MST: A spanning tree T of a weighted connected undirected graph G whose total weight (i.e. $\sum_{e\in T} w(e)$) is minimum



Minimum Spanning Tree (MST)

Theorem: Let $G(V, E)$ be a connected undirected graph. Let A and B be an arbitrary partition of V (i.e. $A \cap B = \emptyset, A \cup B = V$). Let $E_{A,B}$ be all edges of G have one endpoint at A and one endpoint at B. The lightest edge in $E_{A,B}$ (if there is more than one, at least one of them) belongs to MST.

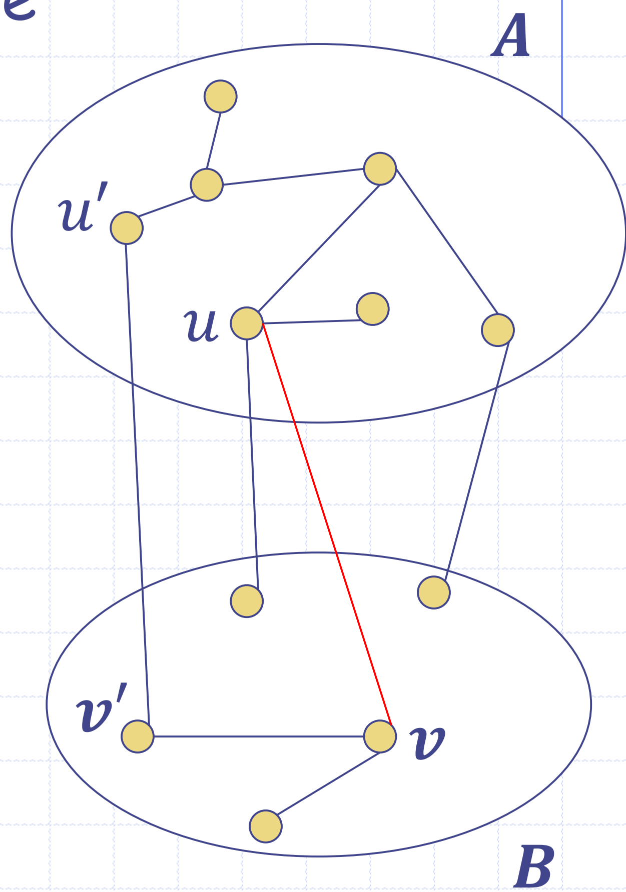
Corollary: The lightest edge incident to a vertex v belongs to MST.

Proof: Let $A = \{v\}, B = V - \{v\}$ in the above theorem.

Minimum Spanning Tree (MST)

The proof of the theorem:

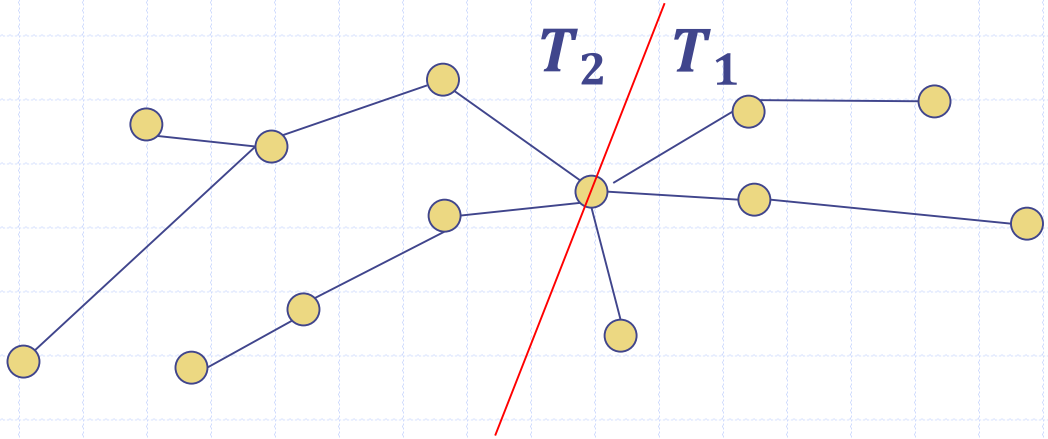
- For the sake of contradiction, assume none of the lightest edges in $E_{A,B}$ exist in MST T.
- Let $e = \{u, v\}$ be one of the lightest edges in $E_{A,B}$ where $u \in A, v \in B$.
- Now consider the graph $T \cup \{e\}$. This graph has a circuit C containing e.
- There should be another edge $e' = \{u', v'\}$ in C such that $u' \in A, v' \in B$.
- We know $w(e) < w(e')$ as e is the lightest edge in $E_{A,B}$.
- Consider $T' = T \cup \{e\} - \{e'\}$. T' is a tree and its weight is smaller than the weight of T which is a contradiction.



Decomposing Trees

Theorem: Suppose T is a tree with n vertices. We can decompose T into two trees T_1 and T_2 such that T_1 and T_2 are disjoint except in a common vertex and

$\left\lceil\frac{n}{3}\right\rceil \leq |T_1|, |T_2| \leq \left\lceil\frac{2n}{3}\right\rceil$ where $|T_i|$ is the size of T_i, i.e. the number of the vertices of T_i .



Remark: This theorem can be used to design divide and conquer algorithms when the input is a tree.

Decomposing Trees

Proof:

- Make tree T rooted by choosing an arbitrary vertex r as the root.
- Compute the size of the subtree rooted at any vertex x, (called it T_x).
- Let u be a vertex with maximum depth satisfying (i) $|T_u| \leq \frac{n}{3}$ and (ii) $|T_{p(u)}| > \frac{n}{3}$ where $p(u)$ is the parent of u
- Let $v = p(u)$ and let $u_1, \dots, u_k$ be the other children of v
- $|T_{u_i}|$ must be at most $\frac{n}{3}$. Otherwise, we can find a vertex in T_{u_i} satisfying (i) and (ii) and its depth is greater than the depth of u.

Decomposing Trees

- If $|T_v| \leq \frac{2n}{3}$, we are done (set $T_1 = T_v, T_2 = (T - T_v) \cup \{v\}$).
- Let $u_0 = u$ and let t be the smallest number such that $\sum_{i=0}^{t} |T_{u_i}| \geq \frac{n}{3}$.
- Since $\sum_{i=0}^{t-1} |T_{u_i}| < \frac{n}{3}$, and $|T_{u_t}| \leq \frac{n}{3}$, then $\frac{n}{3} \leq \sum_{i=0}^{t} |T_{u_i}| < \frac{2n}{3}$.
- Therefore, the tree T_1 consisting of v and subtrees $T_{u_0}, T_{u_1}, \dots, T_{u_t}$ has size at least $\frac{n}{3} + 1$ and at most $\frac{2n}{3}$. Tree $T_2 = (T - T_v) \cup \{v\}$ has size at least $\frac{n}{3}$ and at most $\frac{2n}{3}$

Useful Inequality in B.T.

Theorem: Let T be a rooted binary tree with n vertices. Let $T_u{}^l$ and $T_u{}^r$ be the left subtree and the right subtree of vertex u, respectively. So

$$\sum_{u \in T} \min(|T_u{}^l|, |T_u{}^r|) \le n \log_2 n$$

Remark. This inequality appears in the analysis of some data structures or algorithms like the Union-Find data structure.

Useful Inequality in B.T.

Proof:

- We will prove that each node is counted at most $\log_2 n$ times.
- Suppose node x is counted in terms $\min(|T_{u_1}{}^l|, |T_{u_1}{}^r|), \min(|T_{u_2}{}^l|, |T_{u_2}{}^r|), \dots, \min(|T_{u_k}{}^l|, |T_{u_k}{}^r|)$ where u_{i+1} is the ancestor of u_i.
- Without loss of generality, suppose x is in the subtree $T_{u_1}{}^l$.
- It is easy to see $|T_{u_k}| \geq 2|T_{u_{k-1}}| \geq 2^2|T_{u_{k-2}}| \geq \dots \geq 2^{k-1}|T_{u_1}| \geq 2^k|T_{u_1}{}^l|$.
- We know $|T_{u_k}| \leq n, |T_{u_1}{}^l| \geq 1$
- So, $2^k \leq n$ and therefore $k \leq \log_2 n$