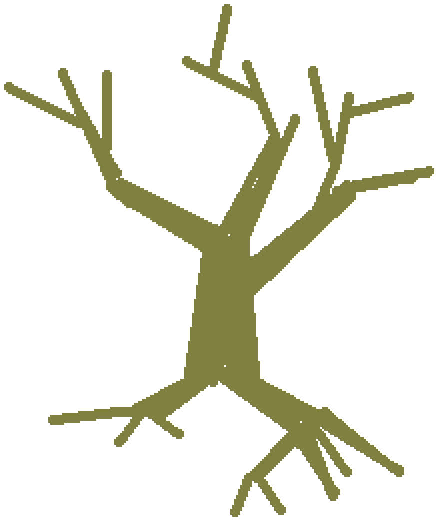
Graph Theory

Trees

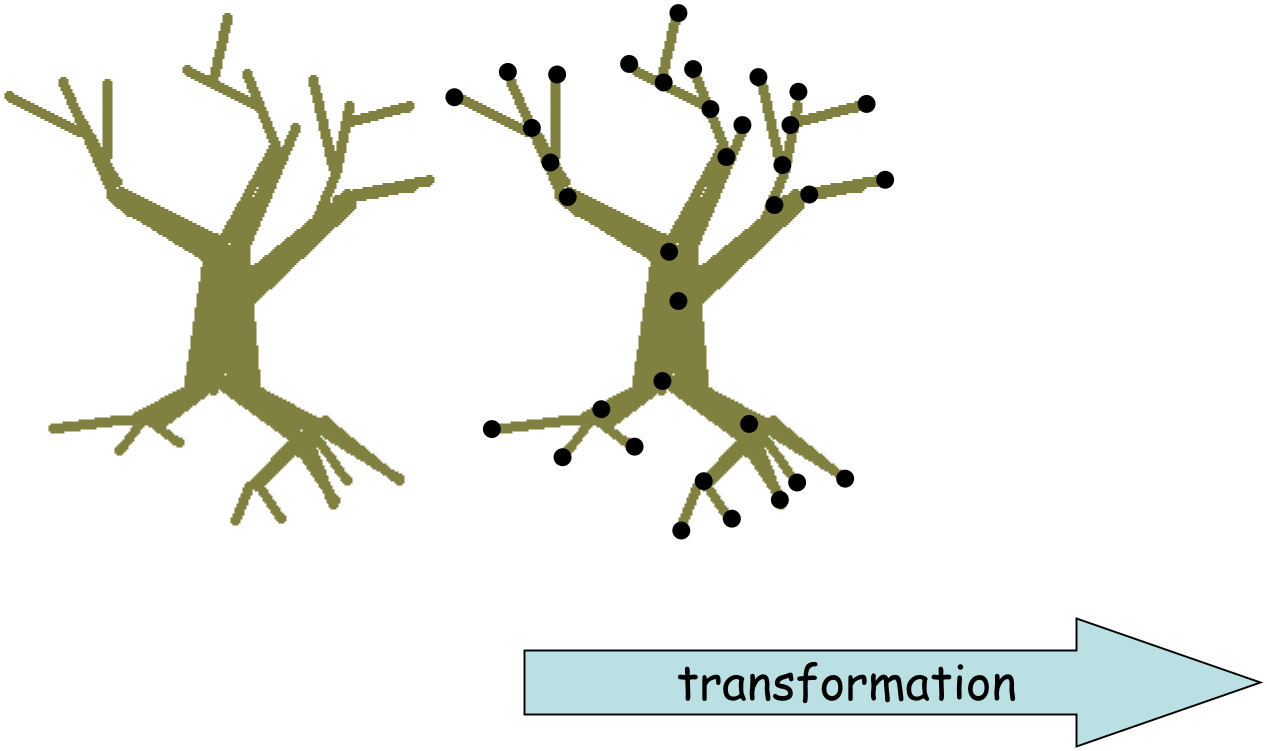
Trees

- A very important type of graph in CS is called a tree:



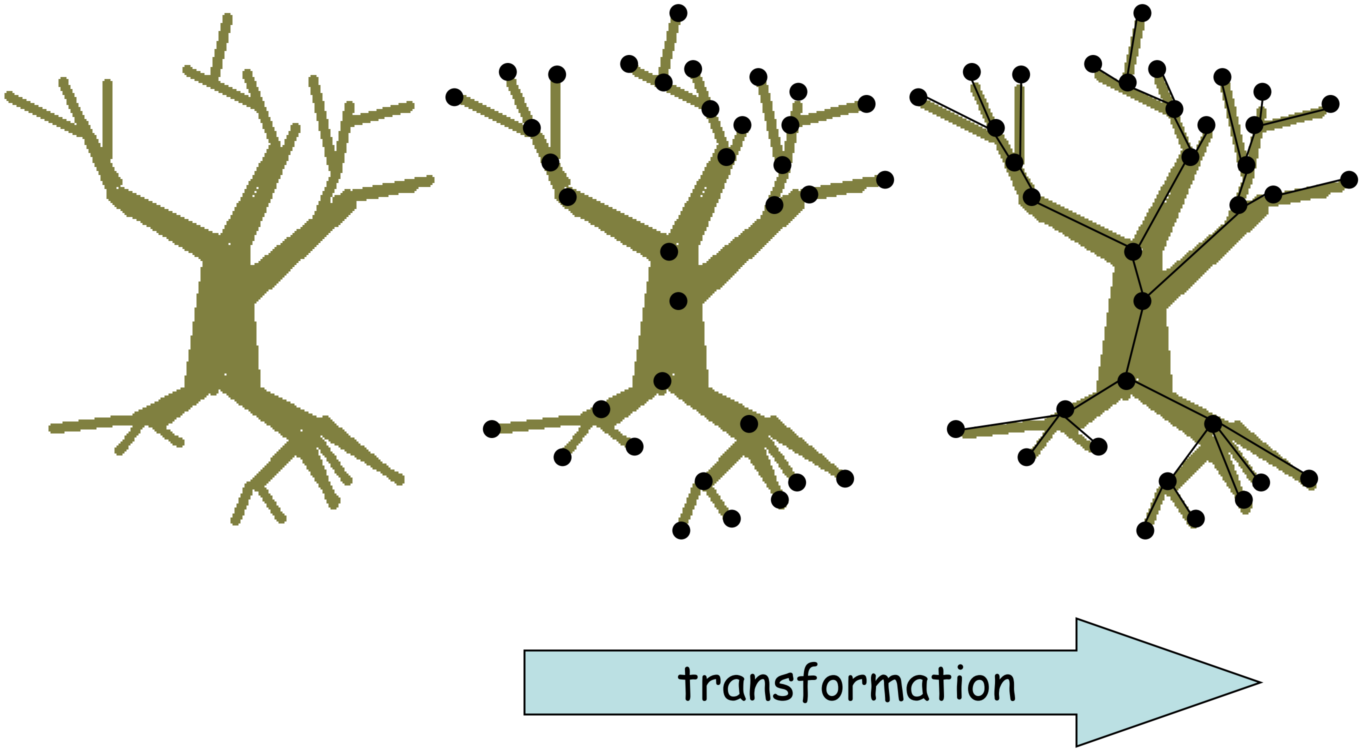
Trees

- A very important type of graph in CS is called a tree:



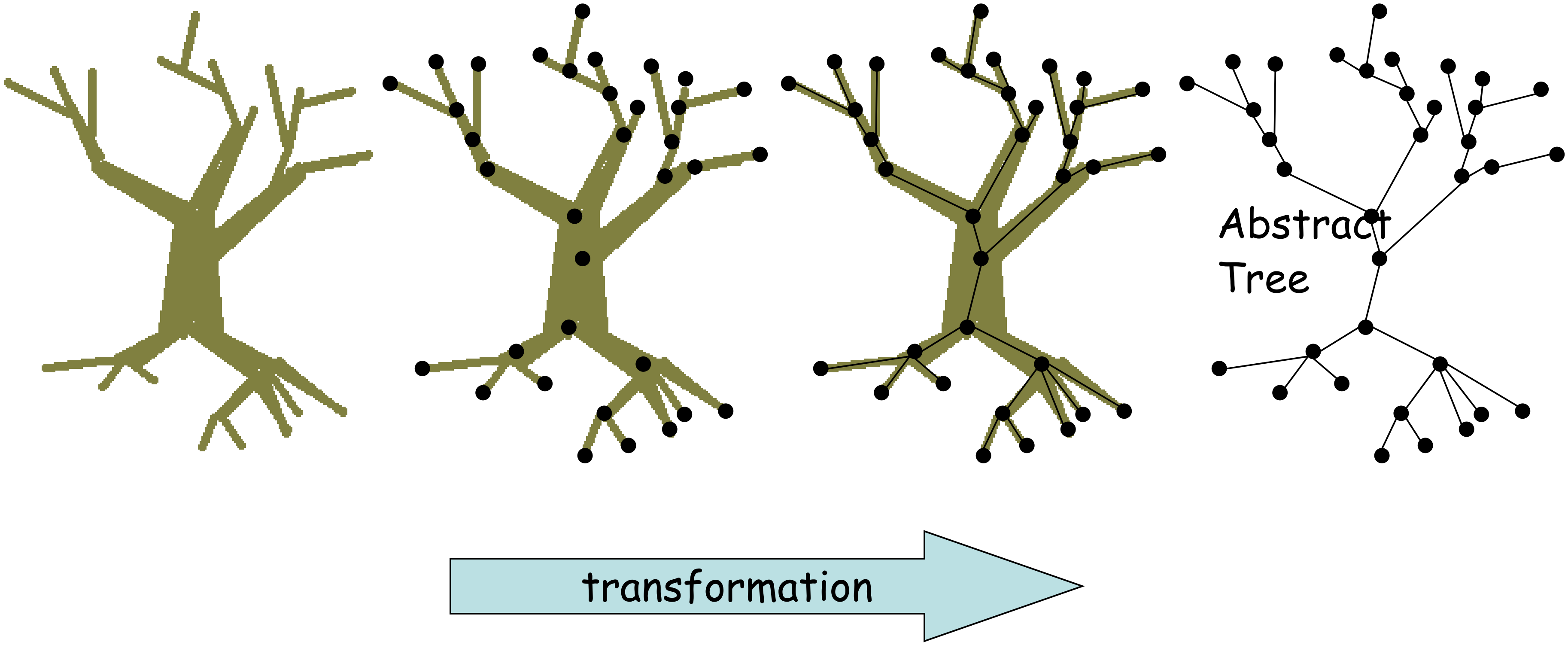
Trees

- A very important type of graph in CS is called a tree:



Trees

- A very important type of graph in CS is called a tree:



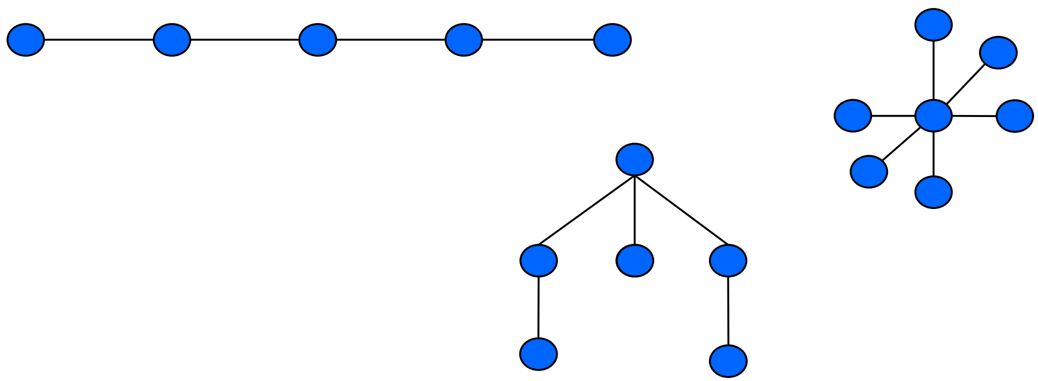
Let us talk about…

Trees

Tree

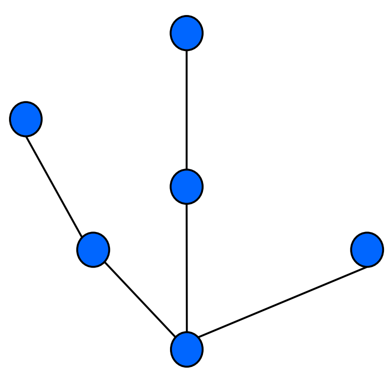
Graphs with no cycles?

A forest.

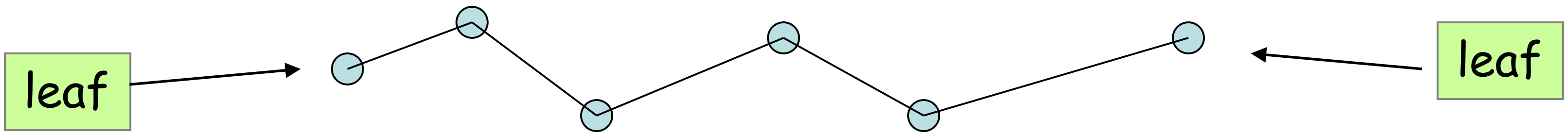


Connected graphs with no cycles?

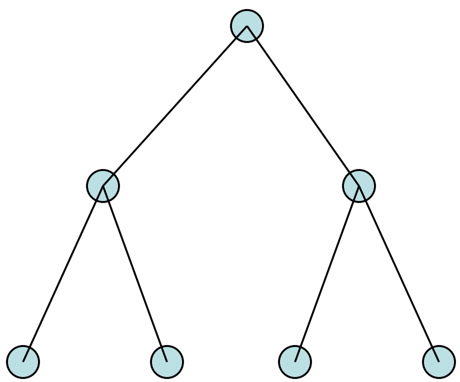
A tree.



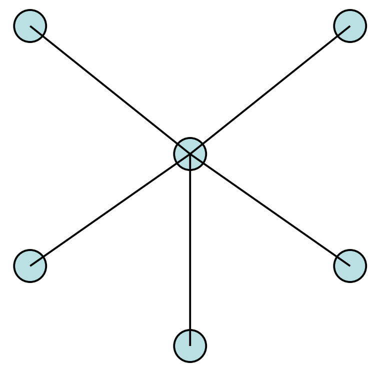
More Trees



A leaf is a vertex of degree 1.



More leaves.



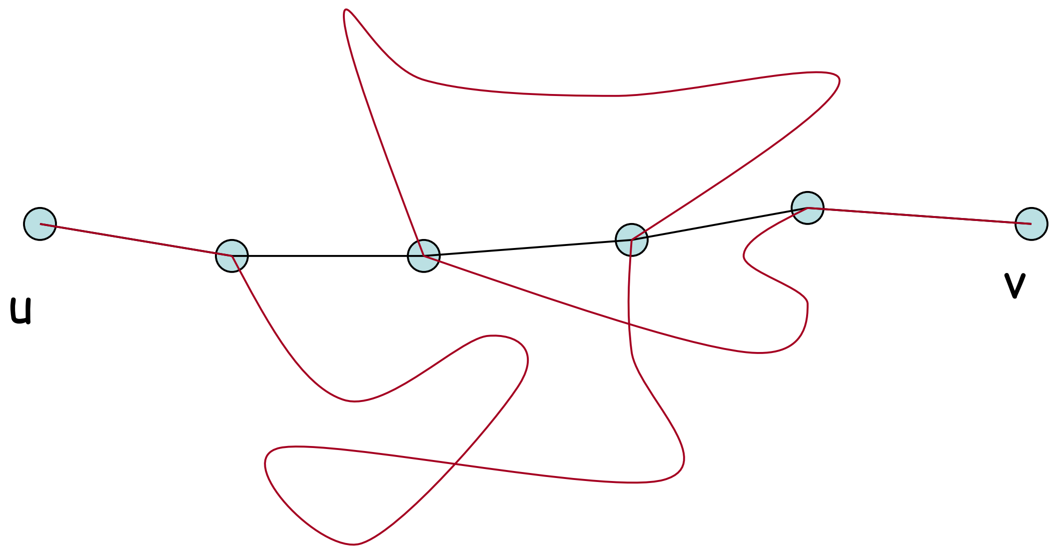
Even more leaves.

Tree Characterization by Path

Definition. A tree is a connected graph with no cycles.

Can there be no path between u and v? NO

Can there be more than one simple path between u and v? NO



This will create cycles.

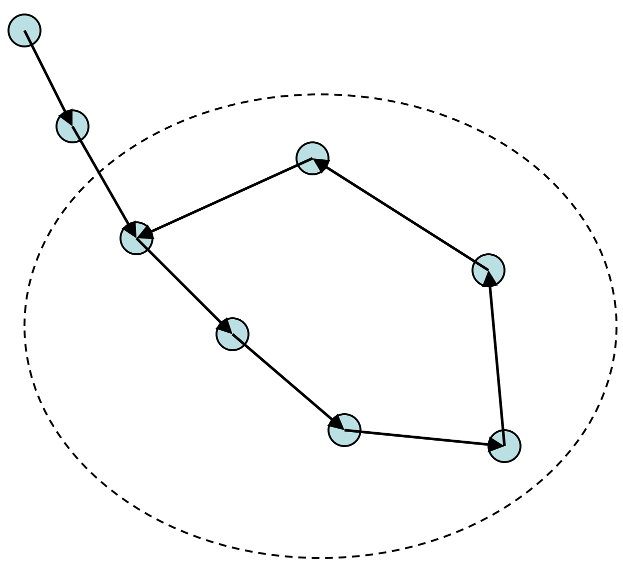
Claim. In a tree, there is a unique simple path between every pair of vertices.

Tree Characterization by Number of Edges

Definition. A tree is a connected graph with no cycles.

Can a tree have no leaves? NO

Then every vertex has degree at least 2.



Go to unvisited edges as long as possible.

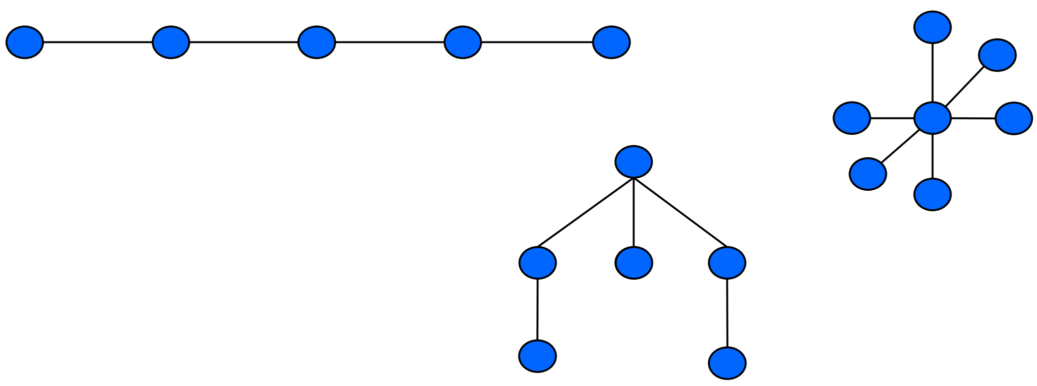
Cannot get stuck,
unless there is a cycle.

Tree Characterization by Number of Edges

Definition. A tree is a connected graph with no cycles.

Can a tree have no leaves? NO

How many edges does a tree have? n-1?



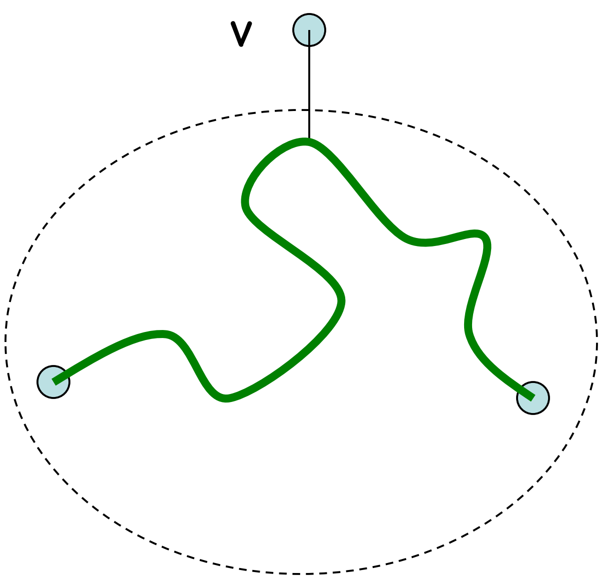
We usually use n to denote the number of vertices,
and use m to denote the number of edges in a graph.

Tree Characterization by Number of Edges

Definition. A tree is a connected graph with no cycles.

Can a tree have no leaves? NO

How many edges does a tree have? n-1?



Look at a leaf v.

Is T-v a tree? YES

1. Can T-v has a cycle? NO
2. Is T-v connected? YES

By induction, T-v has (n-1)-1=n-2 edges.

So T has n-1 edges.

Tree Characterizations

Definition. A tree is a connected graph with no cycles.

Characterization by paths:

A graph is a tree if and only if

there is a unique simple path between every pair of vertices.

Characterization by number of edges:

A graph is a tree if and only if it is connected and has n-1 edges.

(We have only proved one direction.

The other direction is similar and left as an exercise.)

Trees

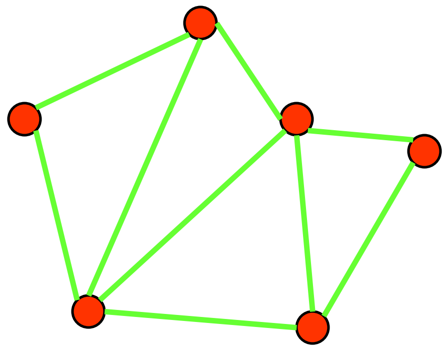
- **Definition:** A **tree** is a connected undirected graph with no simple circuits.
- Since a tree cannot have a simple circuit, a tree cannot contain multiple edges or loops.
- Therefore, any tree must be a **simple graph**.

- **Theorem:** An undirected graph is a tree if and only if there is a **unique simple path** between any of its vertices.

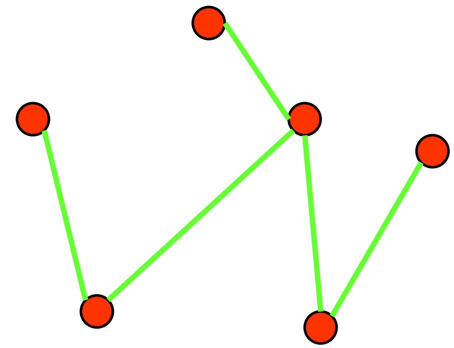
- **Definition:** An undirected graph that does not contain simple circuits and is not necessarily connected is called a **forest**.
- In general, we use trees to represent **hierarchical structures**.

Trees

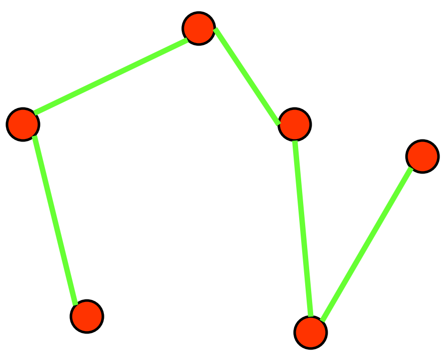
• **Example:** Are the following graphs trees?



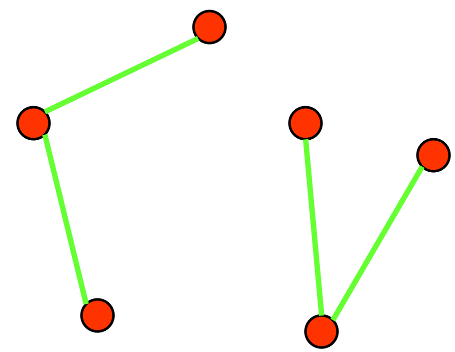
No.



Yes.



Yes.



No.

Spanning Trees

•**Definition:** Let G be a simple graph. A spanning tree of G is a subgraph of G that is a tree containing every vertex of G.

•**Note:** A spanning tree of G = (V, E) is a connected graph on V with a minimum number of edges
(|V| - 1).

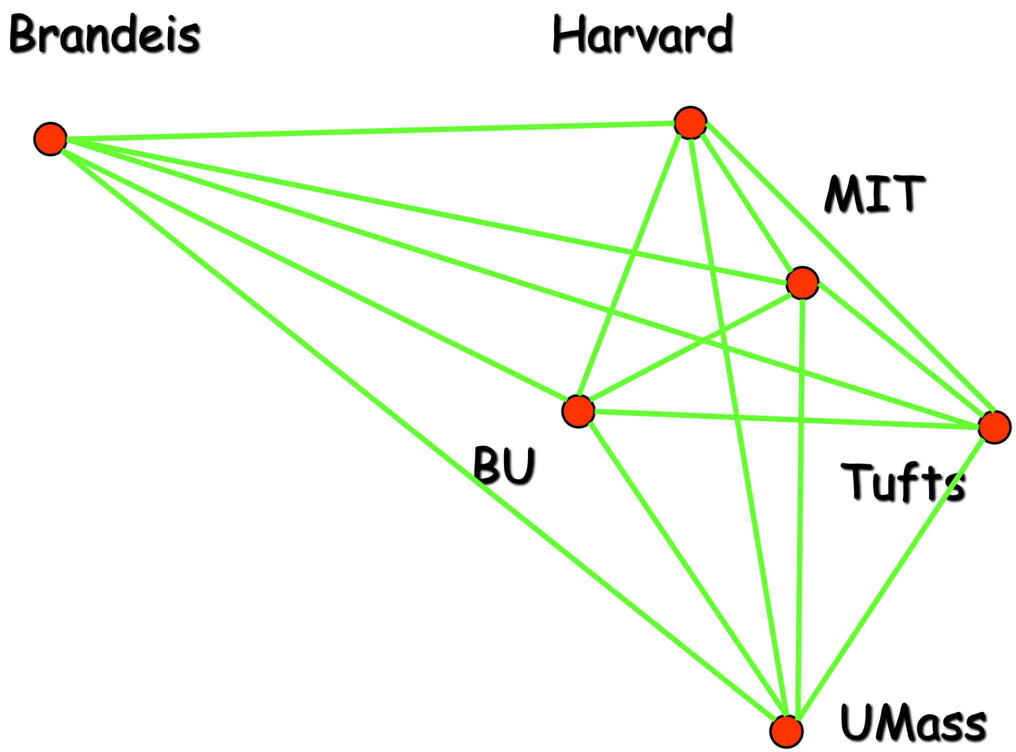
•**Example:** Since winters in Boston can be very cold, six universities in the Boston area decide to build a tunnel system that connects their libraries.

Spanning tree

- A spanning tree in an undirected graph $G(V,E)$ is a subset of edges $T \subseteq E$ that are acyclic and connect all the vertices in V.
- A spanning tree must consist of exactly n-1 edges.
- Suppose that each edge has a weight associated with it. Say that the weight of a tree T is the sum of the weights of its edges $w(T) = \sum_{e \in T} w(e)$
- The minimum spanning tree in a weighted graph $G(V,E)$ is one which has the smallest weight among all spanning trees in $G(V,E)$

Spanning Trees

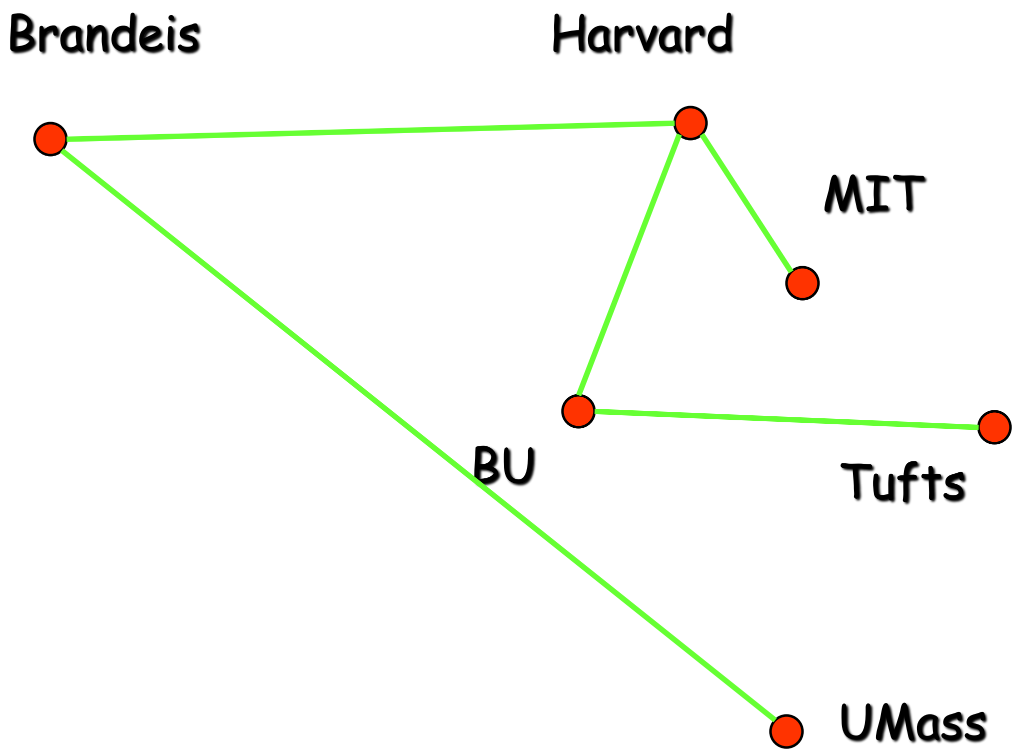
- The complete graph including all possible tunnels:



The spanning trees of this graph connect all libraries with a minimum number of tunnels.

Spanning Trees

- Example for a spanning tree:



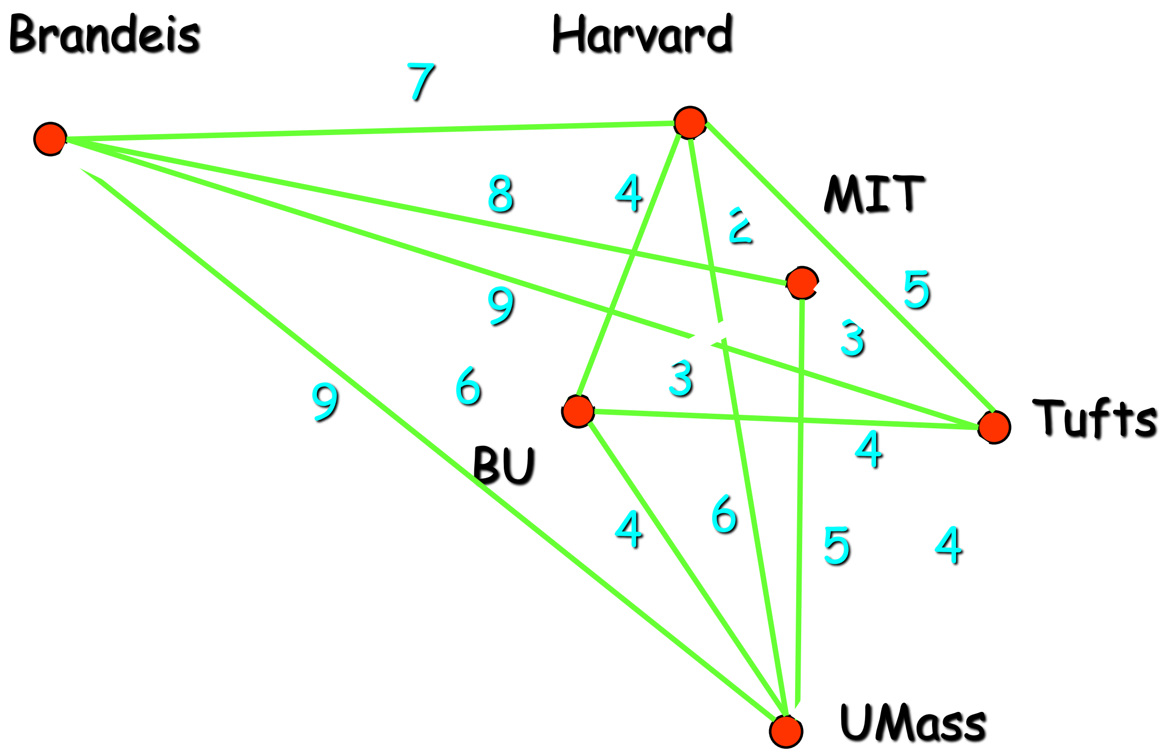
Since there are 6 libraries, 5 tunnels are sufficient to connect all of them.

Spanning Trees

- Now imagine that you are in charge of the tunnel project. How can you determine a tunnel system of **minimal cost** that connects all libraries?
- **Definition:** A **minimum spanning tree** in a connected weighted graph is a spanning tree that has the smallest possible sum of weights of its edges.
- How can we find a minimum spanning tree?

Spanning Trees

- The complete graph with cost labels (in billion $):



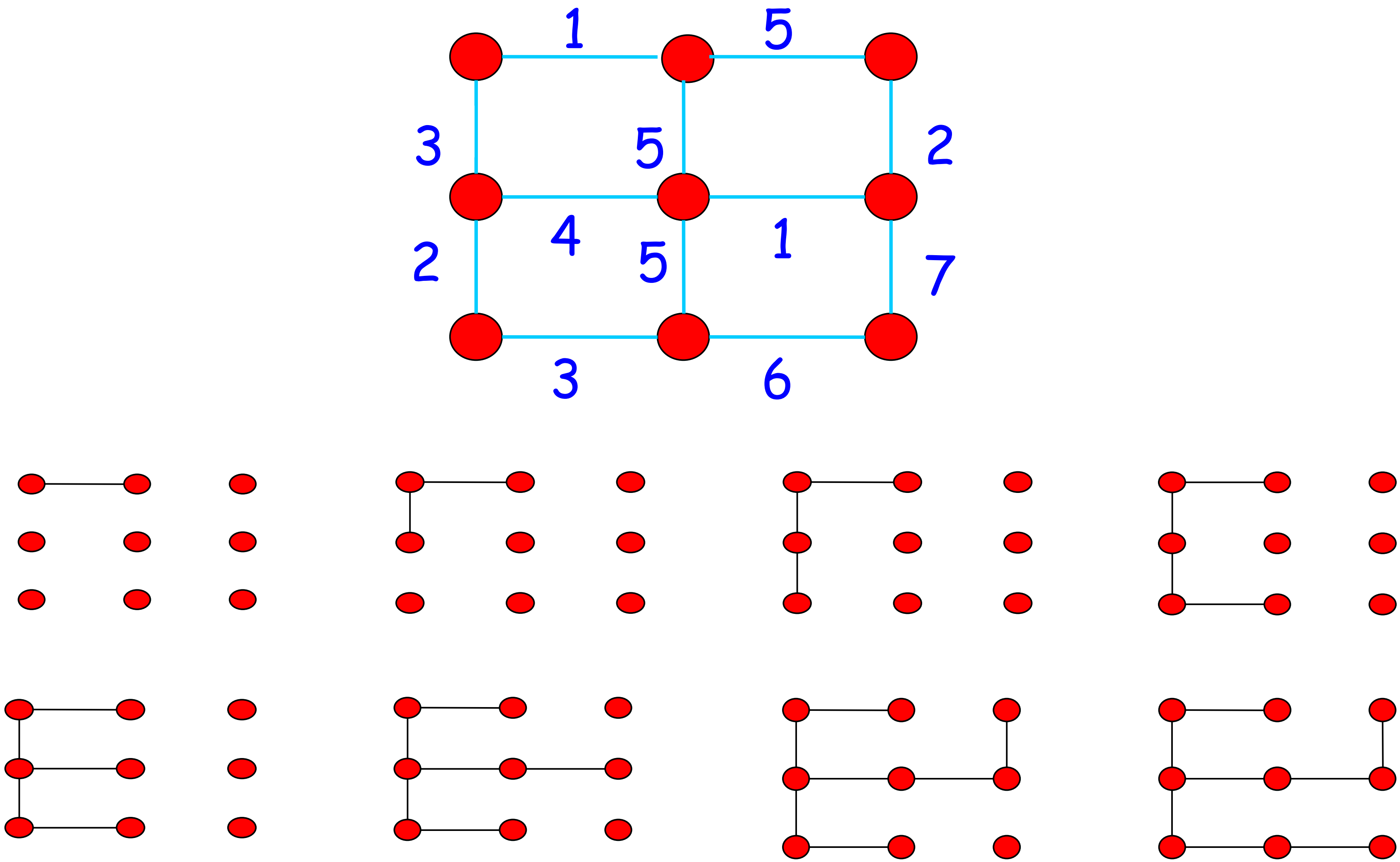
The least expensive tunnel system costs $20 billion.

Spanning Trees

•Prim's Algorithm:

- Begin by choosing any edge with **smallest weight** and putting it into the spanning tree,
- successively add to the tree edges of **minimum weight** that are incident to a vertex already in the tree and not forming a simple circuit with those edges already in the tree,
- stop when (n – 1) edges have been added.

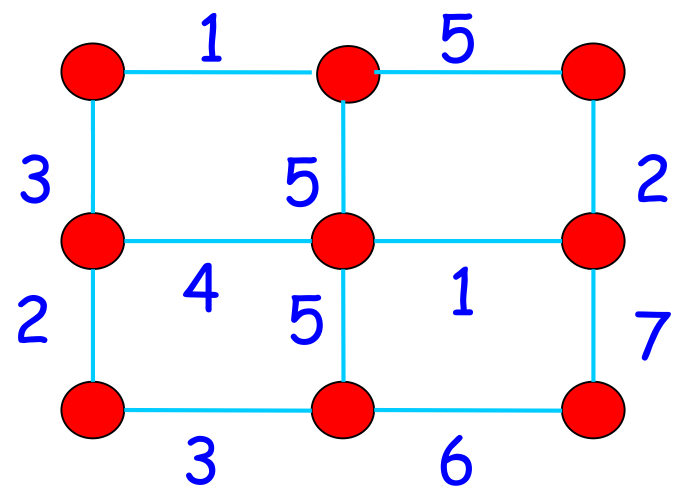
Prim's algorithm

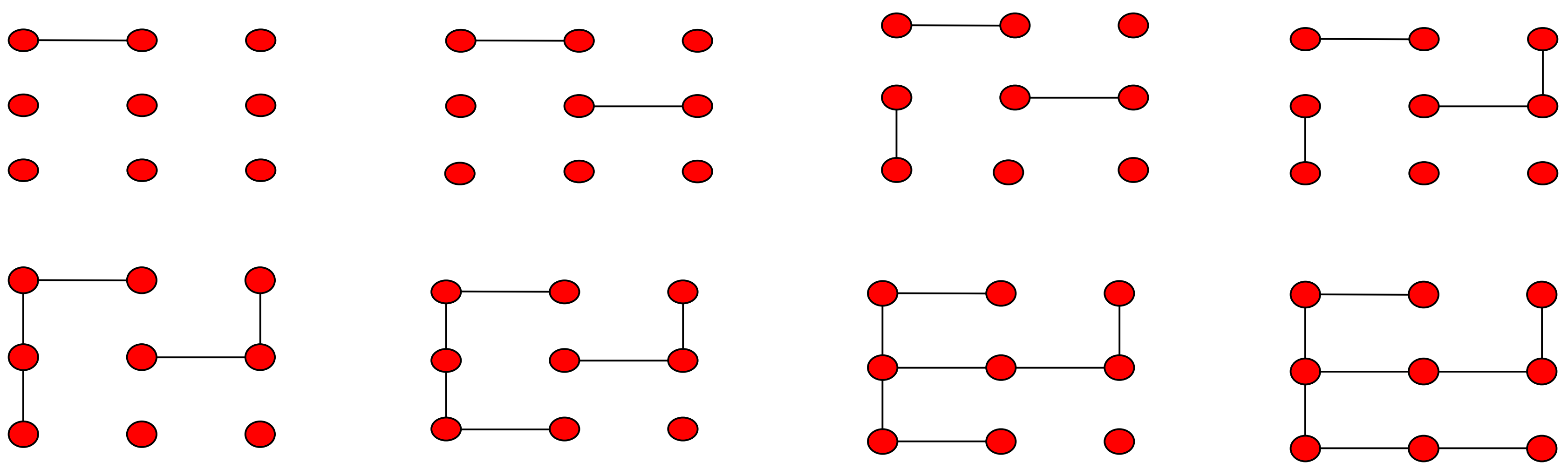


Spanning Trees

- **Kruskal's Algorithm:**
- Kruskal's algorithm is identical to Prim's algorithm, except that it does not demand new edges to be incident to a vertex already in the tree.
- Both algorithms are **guaranteed** to produce a minimum spanning tree of a connected weighted graph.

Kruskal's algorithm





Rooted Trees

- We often designate a particular vertex of a tree as the **root**. Since there is a unique path from the root to each vertex of the graph, we direct each edge away from the root.
- Thus, a tree together with its root produces a **directed graph** called a **rooted tree**.

Rooted Trees

- If v is a vertex in a rooted tree other than the root, the **parent** of v is the unique vertex u such that there is a directed edge from u to v.
- When u is the parent of v, v is called the **child** of u.
- Vertices with the same parent are called **siblings**.
- The **ancestors** of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root.

Rooted Trees

- The **descendants** of a vertex v are those vertices that have v as an ancestor.
- A vertex of a tree is called a **leaf** if it has no children.
- Vertices that have children are called **internal vertices**.
- If a is a vertex in a tree, then the **subtree** with a as its root is the subgraph of the tree consisting of a and its descendants and all edges incident to these descendants.

Rooted Trees

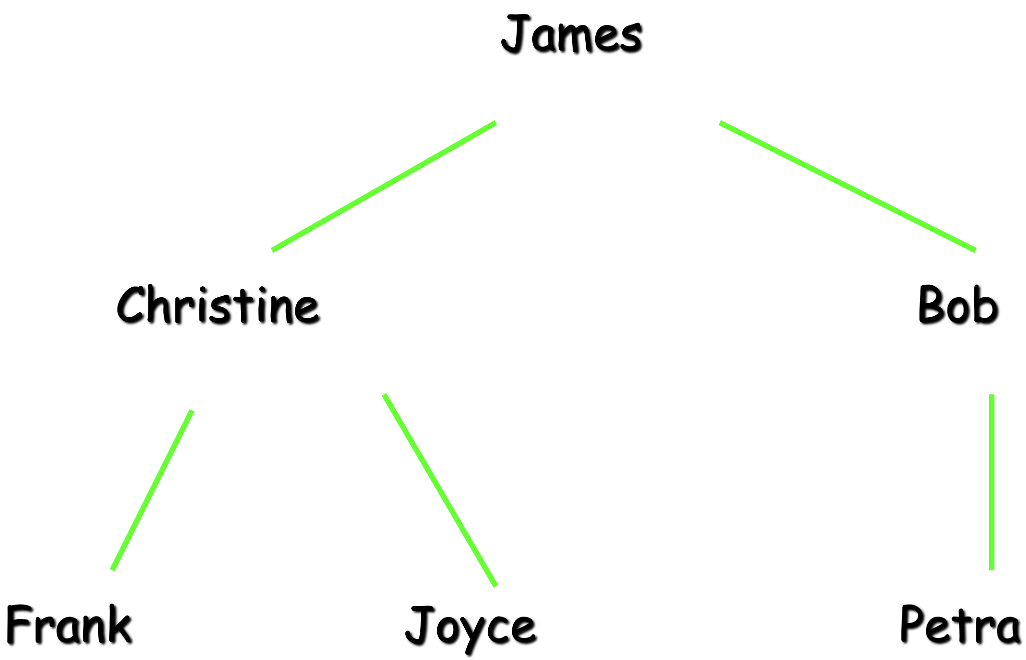
- The **level** of a vertex v in a rooted tree is the length of the unique path from the root to this vertex.

- The level of the root is defined to be zero.

- The **height** of a rooted tree is the maximum of the levels of vertices.

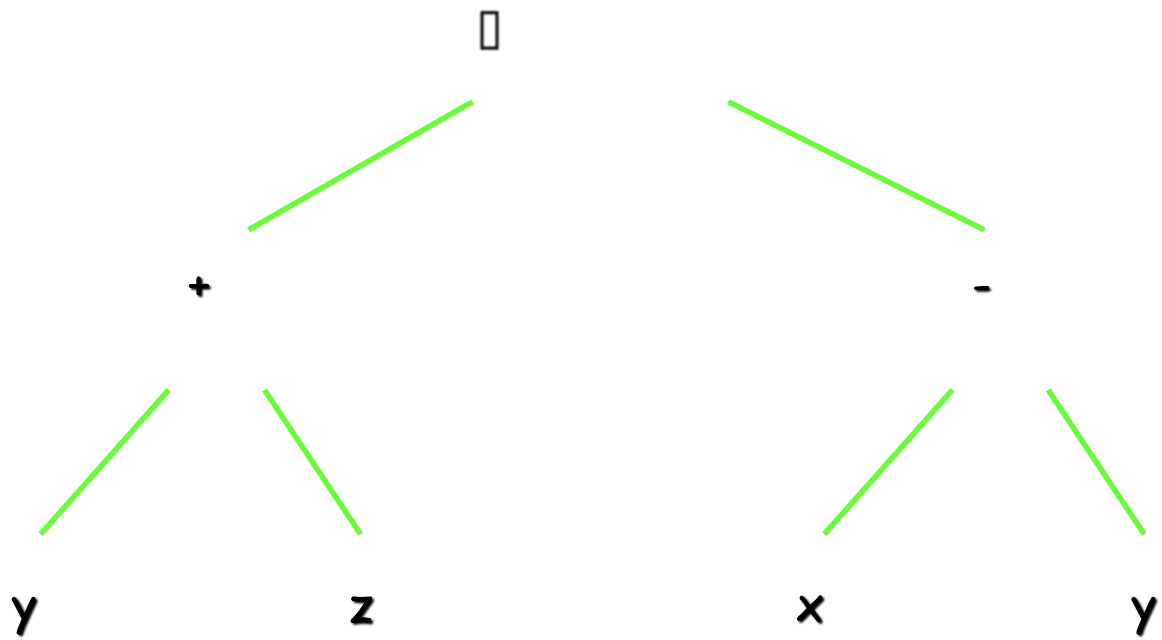
Trees

- **Example I:** Family tree



Trees

• **Example III:** Arithmetic expressions



This tree represents the expression (y + z)   (x - y).

Trees

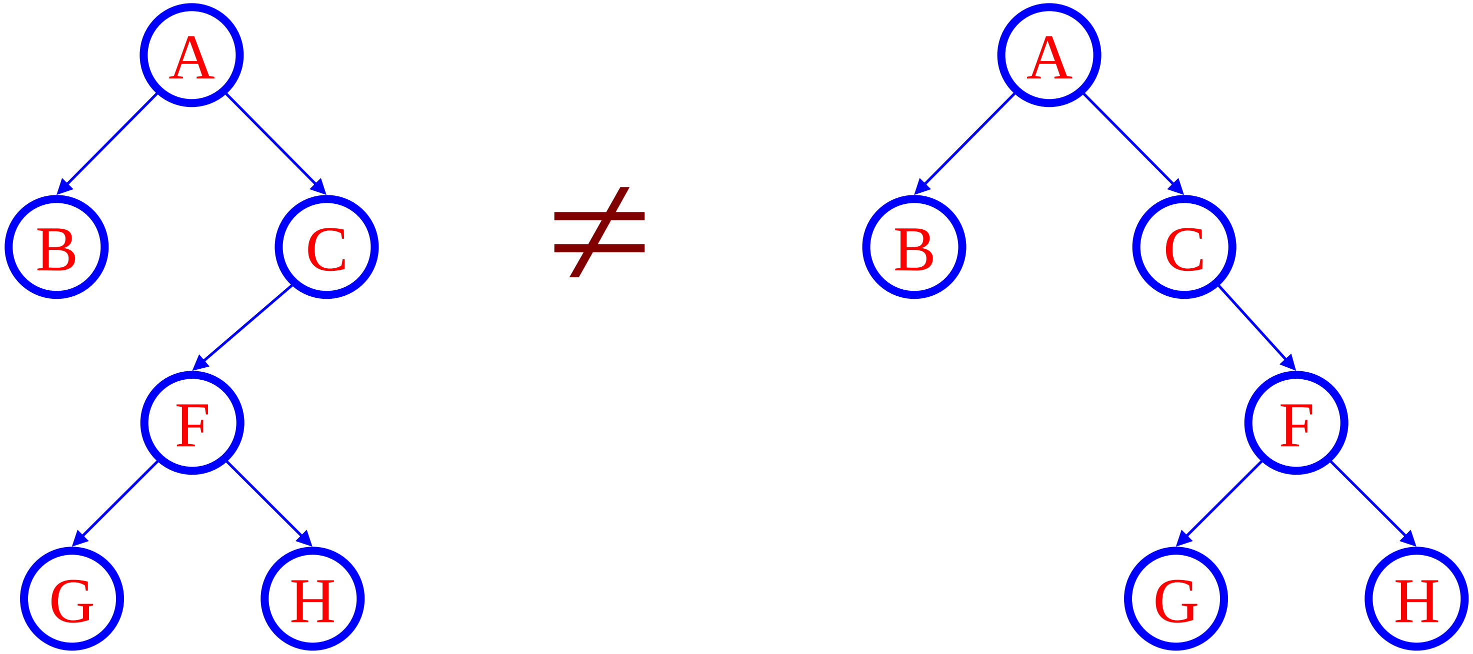
- **Definition:** A rooted tree is called an **m-ary tree** if every internal vertex has no more than m children.
- The tree is called a **full m-ary tree** if every internal vertex has exactly m children.
- An m-ary tree with $m = 2$ is called a **binary tree**.
- **Theorem:** A tree with n vertices has $(n – 1)$ edges.
- **Theorem:** A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.

Binary Trees

- Every node has at most two children
- Most popular tree in computer science
- Given N nodes, what is the minimum depth of a binary tree?
- What is the maximum depth of a binary tree with N nodes?

Binary Trees

- Notice:
- we distinguish between left child and right child



Binary Search Trees

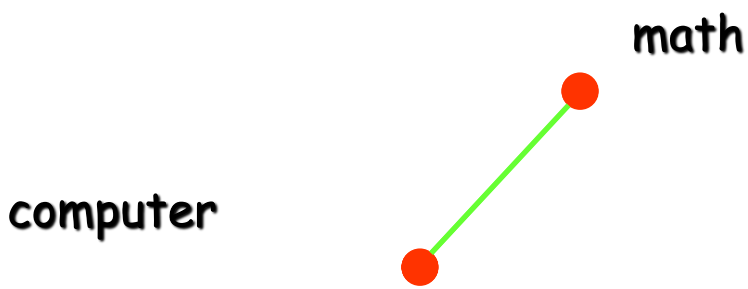
- If we want to perform a large number of searches in a particular list of items, it can be worthwhile to arrange these items in a **binary search tree** to facilitate the subsequent searches.
- A binary search tree is a binary tree in which each child of a vertex is designated as a **right or left child**, and each vertex is labeled with a **key**, which is one of the items.
- When we construct the tree, vertices are assigned keys so that the key of a vertex is both larger than the keys of all vertices in its left subtree and smaller than the keys of all vertices in its right subtree.

Binary Search Trees

•**Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.

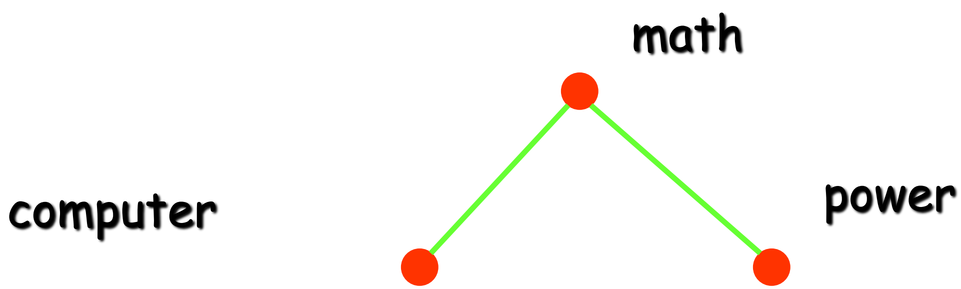
Binary Search Trees

- **Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



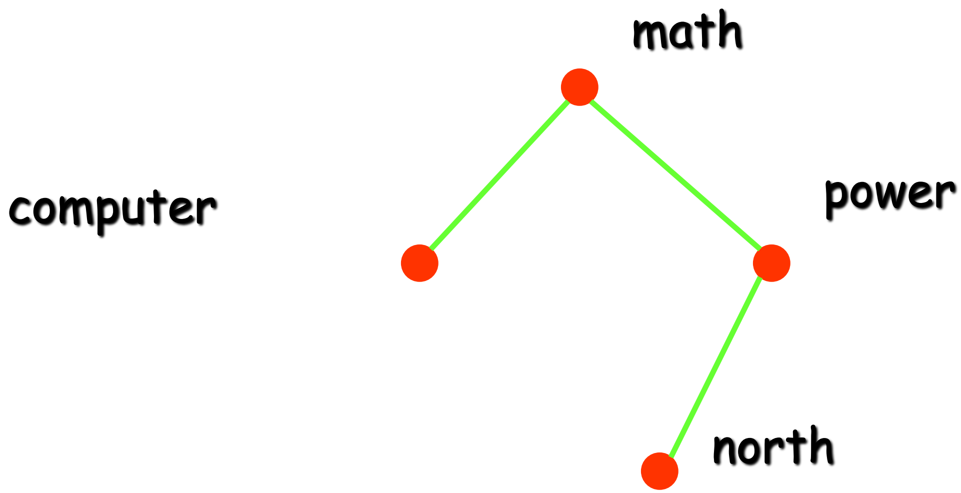
Binary Search Trees

•**Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



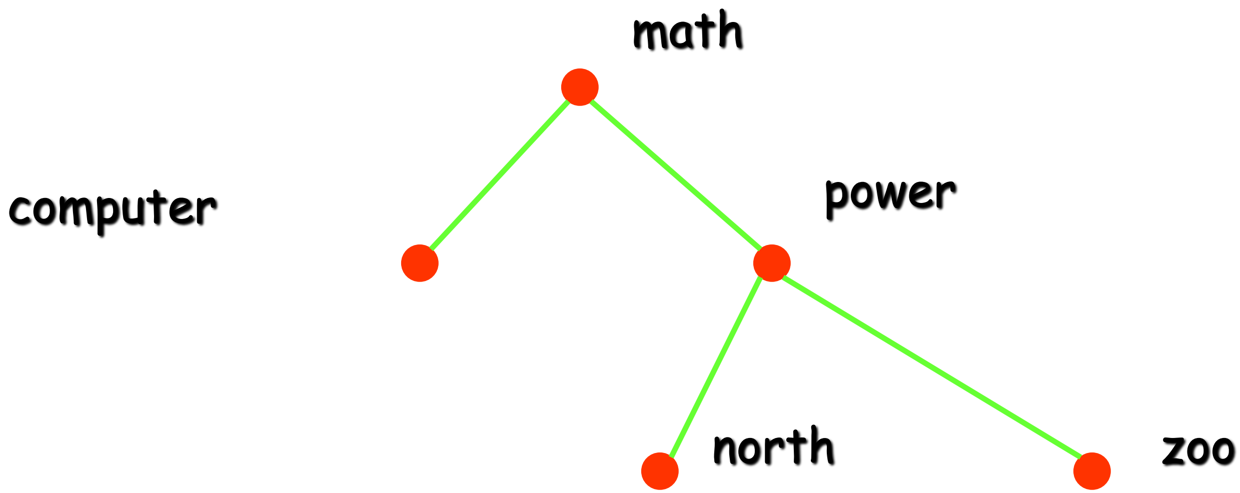
Binary Search Trees

•**Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



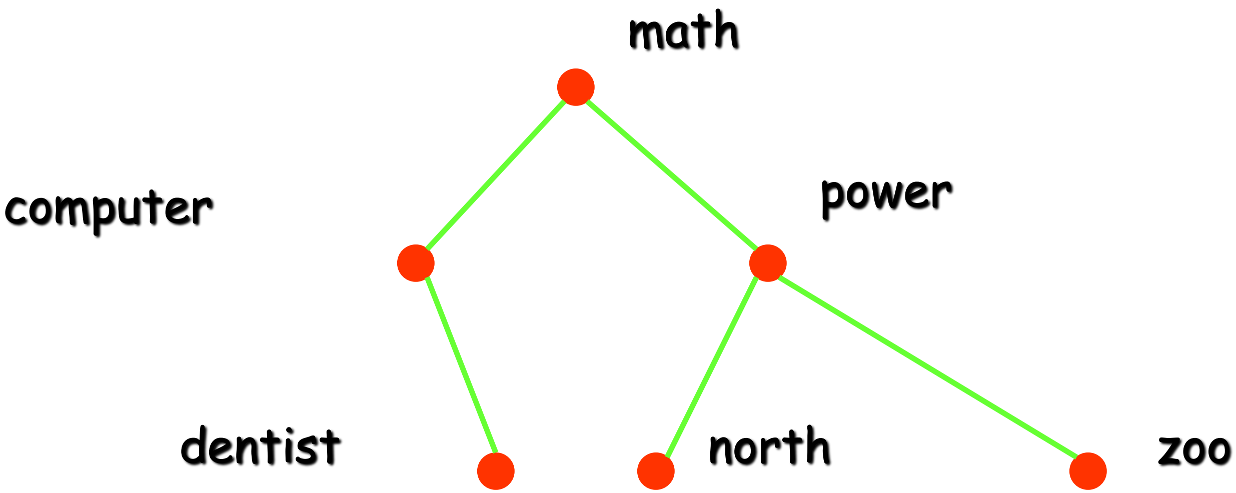
Binary Search Trees

- **Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



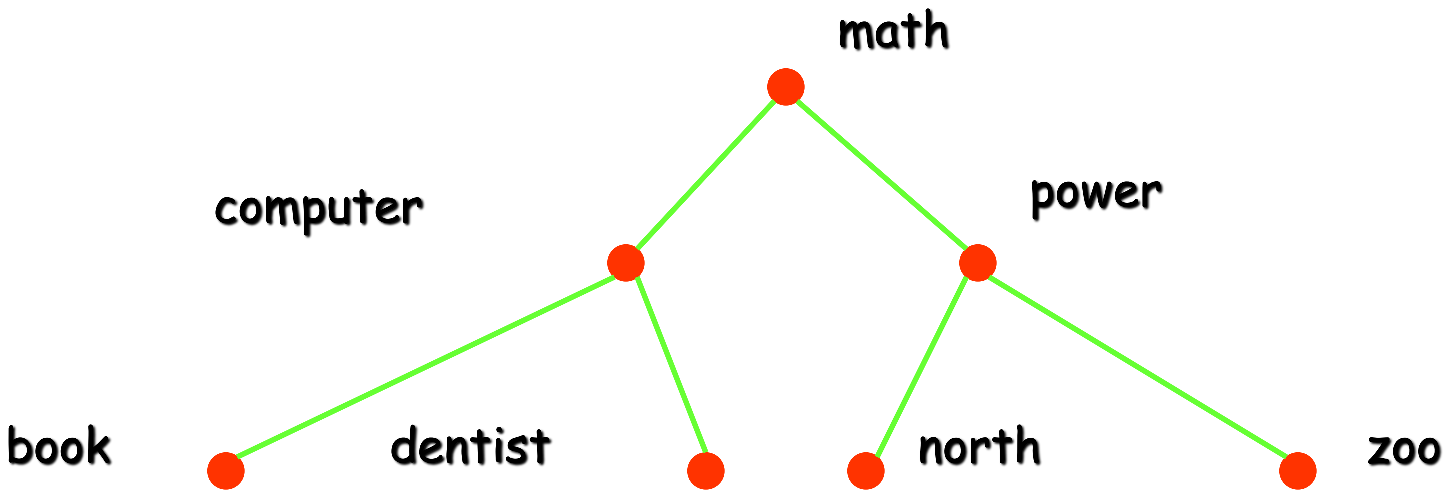
Binary Search Trees

•**Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



Binary Search Trees

- **Example:** Construct a binary search tree for the strings **math, computer, power, north, zoo, dentist, book**.



Binary Search Trees

- To perform a search in such a tree for an item x, we can start at the root and compare its key to x. If x is **less** than the key, we proceed to the **left** child of the current vertex, and if x is **greater** than the key, we proceed to the **right** one.
- This procedure is repeated until we either found the item we were looking for, or we cannot proceed any further.

The End