Topic 6.3
Tree

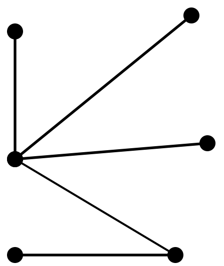
TMA1201 Discrete Structures & Probability
Faculty of Computing & Informatics
Multimedia University

What you will learn in this lecture:

- What is a tree?
- Some definitions and terms
- Binary tree
- Tree traversal
- Spanning tree
- Tree searching

Tree

- A tree is a connected graph with no circuit.
- Examples:



- In computer science, trees are often used to describe a method for traversing a search space.

- **Theorem**: A tree with n >= 1 vertices has (n – 1) edges.
- **Corollary**: Adding any edge to a tree creates a cycle (or circuit).

Try this

- Given a graph G = (V, E) where V = {1, 2, 3, 5, 7} and

  E = {xy|x not equals to y, x divides y or y divides x}.

  Is G a tree?

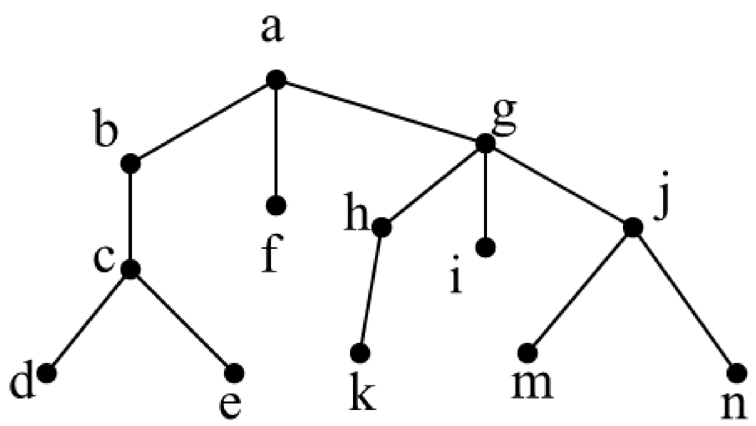
- To see if G is a tree, draw the graph G and refer to the definition of tree……….

- How about when V = {1, 2, 3, 4, 5}?
- How about when V = {2, 3, 4, 5, 6}?

Definitions and terms

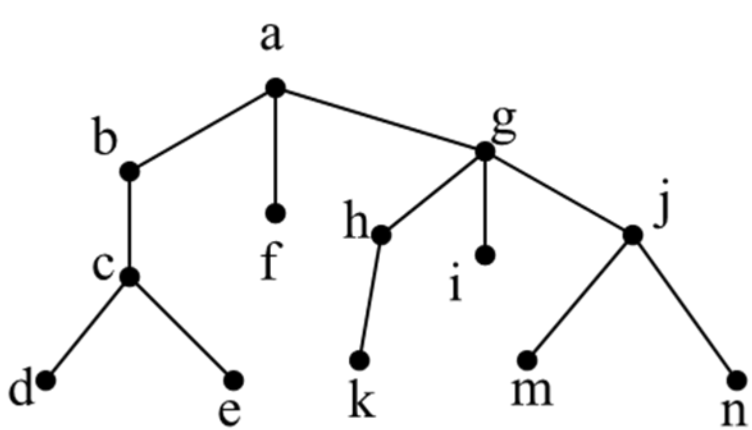
- A **rooted tree** is a tree in which one vertex has been designated as the root and every edge is directed away from the root.
- If *v* is a vertex in a tree other than the root, the parent of *v* is the unique vertex *u* such that there is a directed edge from *u* to *v.*
- When *u* is the parent of *v, v* is called a child of *u.*
- Vertices with the same parent are called siblings.

Example of a rooted tree with vertex a is the root:



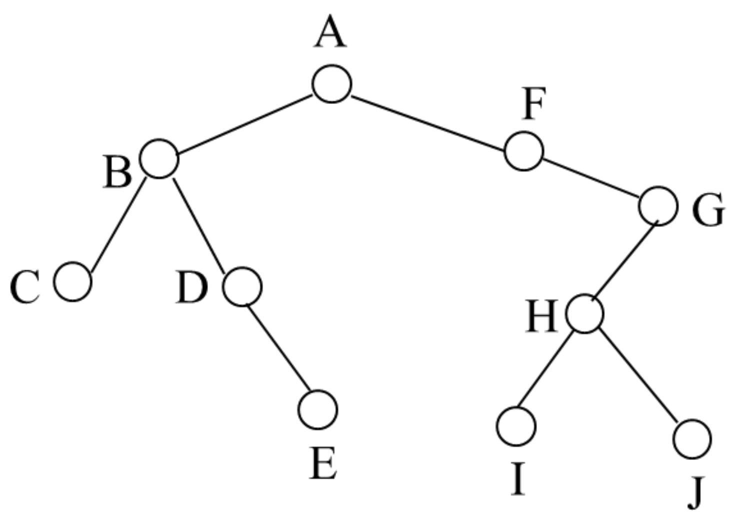
Definitions and terms

- The ancestors of a vertex are the vertices in the path from the root to this vertex.
- 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.
- If *x* is a vertex in a tree, the sub-tree with *x* as its root is the sub-graph of the tree consisting of *x* and its descendants and all edges incident to these descendants
- The **height** of a tree is the maximum of the lengths of simple paths from the root to the leaves.



Binary tree

- A binary tree is a tree, where every node has at most two children.
- One drawn to the left, called the left child, another drawn to the right, called the right child.
- Every node in a binary tree is the root of at most two sub-trees, namely, its left sub-tree, and the right sub-tree.

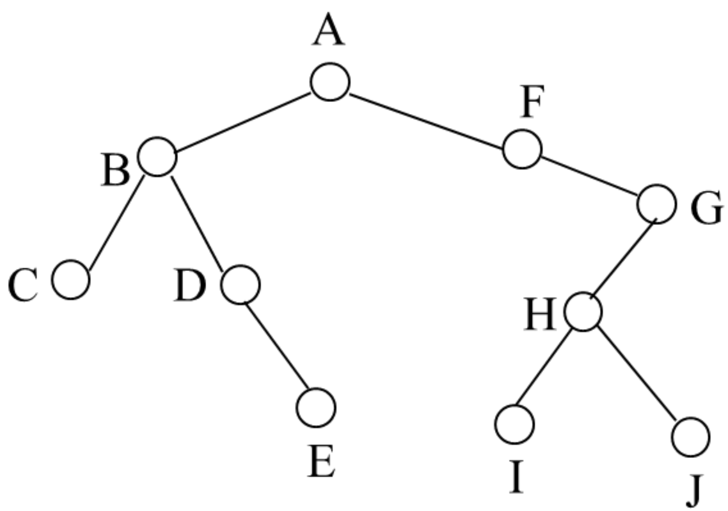


Tree traversal

- Visiting a node in a tree generally means retrieving the data item contained in the node, and sending it to some processes such as, printing.
- There are three ways in which one could traverse all the nodes in a tree.
  - Pre-order traversal
  - In-order traversal
  - Post-order traversal

Pre-order traversal

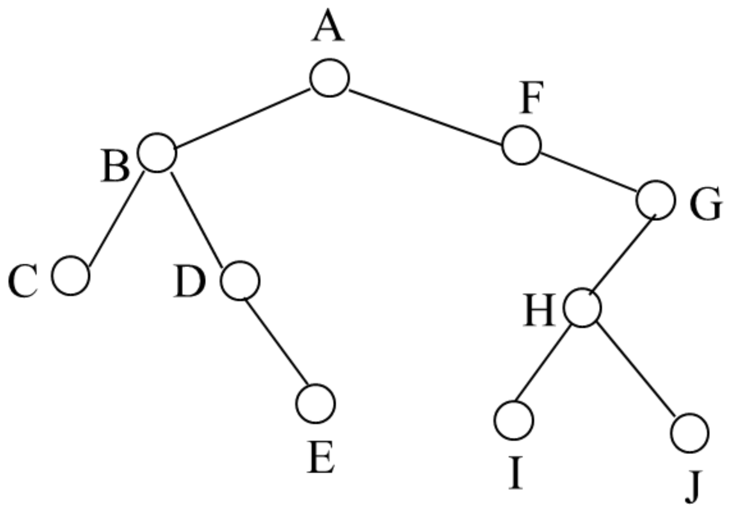
- Visit the current node.
- Traverse the left sub-tree of the current node.
- Traverse the right sub-tree of the current node.



The order using pre-order traversal is: ABCDEFGHIJ

In-order traversal

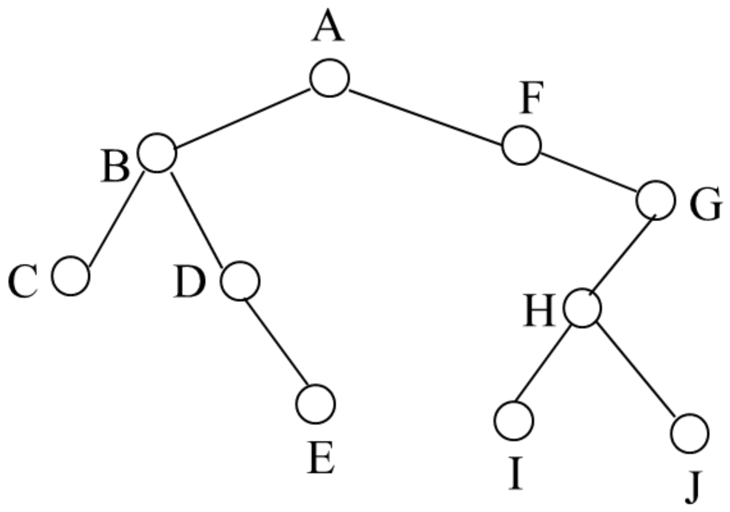
- Traverse the left sub-tree of the current node.
- Visit the current node.
- Traverse the right sub-tree of the current node.



The order using in-order traversal is: C B D E A F I H J G

Post-order traversal

- Traverse the left sub-tree of the current node.
- Traverse the right sub-tree of the current node.
- Visit the current node.

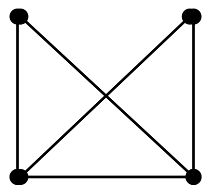


The order using post-order traversal is: C E D B I J H G F A

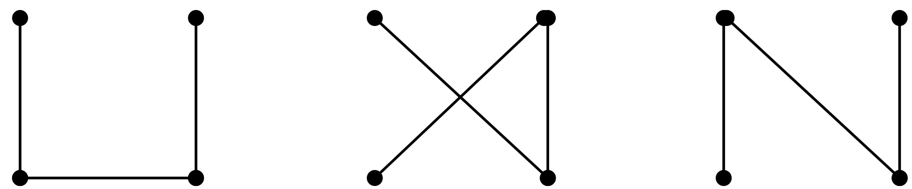
Spanning tree

- A spanning tree of a graph G = (V, E), is a subgraph of G that is a tree containing every vertex of G.

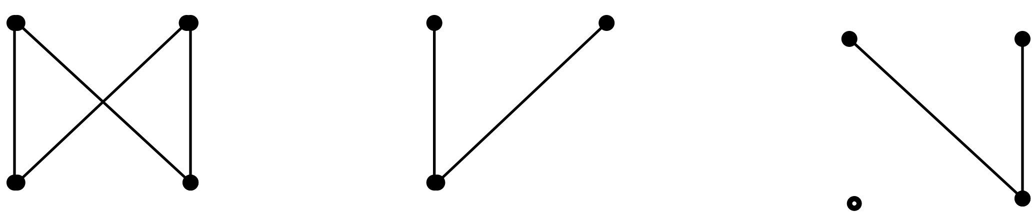
- Given a graph:



- Examples of spanning trees:



- Examples of subgraphs that are not spanning tree



Spanning tree

Theorem:

- Any connected graph G contains a spanning tree.

- We can proof this theorem by induction on the number of edges (for info)
- Inductive Base: If G_1 has one edge, then the edge with its endpoints is a spanning tree of G_1
- Inductive Hypothesis: Assume that any connected graph with number of edges <= k, G_k , has a spanning tree.
- Inductive Step: We need to prove that we can find a spanning tree for any connected graph with k+1 edges, G_{k+1}.

  If G_{k+1} has no circuit, then G_{k+1} is a spanning tree itself.

  If G_{k+1} has a circuit C, we can remove any edge e from C and $G_{k+1} – \{e\}$ is still connected. Since $G_{k+1} – \{e\}$ has one edge less, it contains a spanning tree T by induction, where T is also a spanning tree for G_{k+1}.

Finding a spanning tree from a graph

- **Depth-first search (DFS)** and **breadth-first search (BFS)** can be used to find a spanning tree of a graph or to search a tree for a particular data.
- For DFS we start by selecting any vertex V_0, and add it to a stack S. It is a Last In First Out (LIFO) process. A helpful analogy is to think of a stack of books; you can remove only the top book, also you can add a new book on the top.
- For BFS we start by selecting any vertex V_0, and add it to a queue Q. It is a First In First Out (FIFO) process. An excellent example of a queue is a line of customer in front of a bank counter. New additions to a line are made to the back of the queue, while removal (or serving a queue) happens at the front of a queue.
- BFS and DFS are widely used in games design, such as in tic-tac-toe and chess games.

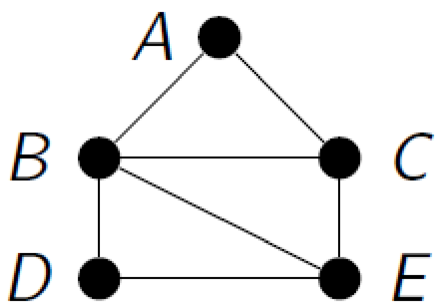
Depth-first search (DFS)

- The basic idea of depth-first search is to go as deeply as possible into a path before reaching out to other vertices.
- To find a spanning tree by DFS of a graph, start by selecting any vertex V_0, and add it to a stack S.
- The stack is processed as follows:
  - If vertex V is at the top of the stack S, we find its neighbours (that haven't been discovered before) and add them to the stack.
  - A vertex at the top of the stack is removed from the stack if all of its neighbours have been discovered and added to the stack.

Depth-first search (DFS)

Example:

Use DFS to find a spanning tree for this graph. Assume vertex C is the root and the selection of neighbouring vertex in each iteration is based on the vertex with the lowest alphabetical order.



Solution:

No	Neighbouring	Stack	Tree
1	-	C	-
2	A,B,E	CA	CA
3	B	CAB	CA,AB
4	D,E	CABD	CA,AB,BD
5	E	CABDE	CA,AB,BD,DE
6	-	CABD	CA,AB,BD,DE
7	-	CAB	CA,AB,BD,DE
8	-	CA	CA,AB,BD,DE
9	-	C	CA,AB,BD,DE
10	-	-	CA,AB,BD,DE

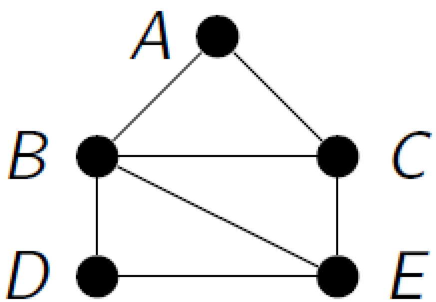
Breadth-first search (BFS)

- The basic idea of breadth-first search is to reach out to as many vertices as possible before penetrating deep into the next level.
- To find a spanning tree by BFS of a graph, start by selecting any vertex V_0, and add it to a queue Q.
- The queue is processed as follows:
  - If vertex V is at the front of the queue Q, we find its neighbours (that haven't been discovered before) and add them to the queue.
  - A vertex at the front of the queue is removed from the queue if all of its neighbours have been discovered and added to the queue.

Breadth-first search (BFS)

Example:

Use BFS to find a spanning tree for this graph. Assume vertex D is the root and the selection of neighbouring vertex in each iteration is based on the vertex with the lowest alphabetical order.



Solution:

No	Neighbouring	Queue	Tree
1	-	D	
2	B,E	DB	DB
3	E	DBE	DB,DE
4	-	BE	DB, DE
5	A,C	BEA	DB, DE, BA
6	C	BEAC	DB, DE, BA, BC
7	-	EAC	DB, DE, BA, BC
8	-	AC	DB, DE, BA, BC
9	-	C	DB, DE, BA, BC
10	-	-	DB, DE, BA, BC

Summary

Materials covered in this lecture:

- Introduction to tree and its related terms, binary tree, spanning tree.
- Tree traversal.
- Apply BFS and DFS to find a spanning tree from a graph.

Exercise 1

- Given a graph G = (V, E) where V = {0, 2, 4, 6, 8, 10} and
  E = {xy|x not equals to y, x divides y or y divides x}.
  Is G a tree? Explain your answer.

Exercise 2

Find the spanning tree from the following graph by using

1) DFS
2) BFS

Assume D is the root and the selection of neighbouring vertex in each iteration is based on the vertex with lowest alphabetical order.