Semester Final

Lecture - 6 (slide):

Q. Why use ~~s~~counting?

⇒ Counting is used to determine the complexity of algorithms. ~~Furthermore~~ Counting techniques are used extensively when probabilities of events are computed.

(multiplication Rule)

The Product Rule: If a task can be performed in $\cancel{n}$ n_1 ways and a second task ~~is~~ can be performed in n_2 ways, and these tasks are independent, then total number of ways to perform both tasks is $n_1 \times n_2$

$\therefore$ Total ways $= n_1 \times n_2$

The Sum Rule (Addition Rule:) If a tasks can be performed in ~~m~~ n_1 ways or in n_2 ways but not both simutaneously, then the total number of ways to perform the task is $m+n$:

$\therefore$ Total ways $=$ ~~$m+n$~~ n_1+n_2

Key Difference:

#Sum rule: sum rule applies when events are mutually exclusive (either one task or another can happen, not both)

 # Product Rule: Product rule applies when events are independent (you perform both tasks in sequence, with each choice being independent of the other).

The Pigeonhole Principle; If k is a positive integer and $k+1$ or more objects are placed into k boxes, then there is at least one box containing two or more of the objects.

The Generalized Pigeonhole Principle:

If N objects are placed into k boxes, then there is at least one box containing at least $\lceil N/K \rceil$ objects.

Lecture - 7 (Permutation & Combination)

Permutations: Any arragement of a set of n objects in a given order is called a permutation of the object (taken all at a time).

Corollary: There are $n!$ permutations of n objects. (taken all at a time).

r-permutations: An ~~re~~ r-permutations is a way to arrange or order r items chosen from a set of n distinct items where $r \leq n$.

The number of r-permutations:

$$P(n,r) = \frac{n!}{(n-r)!}$$

r- combination: An r-combination is way of choosing r items from a set of n distinct items without considering the order.

Number of r-combinations is given by:

$$C(n, r) = \frac{n!}{r!\,(n-r)!}$$

Lecture - 8 (Graph)

Graph! A graph is a mathematical structure consisting with vertices (nodes) and edges to show relationships or connections between things.

Vertices : The "dots" in the graph
Edges : The "lines" that connect the dots.

Simple Graph: A graph in which each edge connects two different vertices and where no two edges connect the same pair of vertices is called a simple graph.

Multigraphs: Graphs that may have multiple edges connecting the same vertices are called multigraphs.

Pseudographs: Edges that connect vertex to itself called loops. A graph with loop (self-loop) is called pseudograph.

Undirected Graph: An undirected graph is a graph where the edges have no direction. The edges simply connect with two vertices.

Directed Graph: A directed graph is a graph where each edge has a direction. The edges point from one vertex to another.

Degree in an Undirected Graph: The degree of a vertex in an undirected graph is the number of edges connected to it considering the loops twice.

Degree (Directed Graph): In directed graph where edges have direction, each vertex has two degrees:

In-degree ($deg^-(v)$): number of edges coming into the vertex

Out-degree ($deg^+(v)$): number of edges going out of the vertex.

$$\boxed{\text{Total degree} = deg^-(v) + deg^+(v)}$$

Lecture-9 (Graph)

Handshaking Theorem: "The Sum of degrees over all the vertices equals twice the number of edges."

$$\sum \deg(v) = 2e$$ where,

v = vertices

e = edges

considerable even if multiple edges and loops are present.

Bipartite Graph: A bipartite graph, is a graph whose vertices can be divided into two disjoint sets such that every edge connects a vertex in one set to a vertex in the other set.

Lecture-10 (Graph-Connectivity)

Path: A path in a graph is a continuous way of getting from one vertex to another by using a sequence of edges.

Simple Path: A simple path or circuit is simple if it does not contain the same edge more than once. (duplicate vertices are allowed)

A cycle: A cycle or circuit is a path which starts and ends at the same vertex.

Lecture-11 (Graph)

Euler Path & Circuit

Euler Path: A path in a graph that uses every edges exactly once. A graph has an Euler path if it has exactly two vertices with odd degrees.

Euler Circuit: A circuit that uses every edges exactly once. A graph has an Euler circuit if all vertices have even degrees.

Hamilton Path & Circuit

Hamilton Path: A path in a graph that visits every vertex exactly once but does not need to cover every edge.

Hamilton Circuit: A circuit (a path that starts and ends at the same vertex) that visits every vertex exactly once.

Weighted Graph: Graphs that have a number assigned to each edge are called ~~wi~~ weighted graphs.

Lecture-12 (Tree_1)

Tree: A tree is a connected undirected graph with no simple circuit, no multiple edges and ~~to~~ no loops.

An undirected graph is a tree if and only if there is unique simple path between any two of its vertices.

Forest: A forest is a disjoint union of trees, meaning it is a collection of one or more trees that are not connected to each other.

A vertex with no ~~children~~ children is called a leaf.

Vertices with children are called internal vertices

Properties of Trees: i) A tree with n vertices has $n-1$ ~~degree~~ edges.

ii) An full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.

iii) A rooted m-ary tree of height h is called balanced if all leaves are at levels h or $h-1$.

Full Binary Tree: A full binary tree is a binary tree in which each node is either a leaf node or has degree 2.

Complete Binary Tree: A complete binary tree is a full binary tree in which all leaves have the same depth.

Traversal: A traversal algorithm is a procedure for systematically visiting every vertex of an ordered rooted tree.

3 common traversals are:

i) Pre-order

ii) In-order

iii) Post-order