Graph Traversal – Summary

1. Definition

Graph Traversal is the process of visiting all the vertices and edges in a graph systematically. Traversal is used to explore a graph, search for nodes, or solve graph-based problems.

2. Types of Graph Traversal

2.1 Depth-First Search (DFS)

Visits a vertex and then recursively explores as far as possible along each branch before backtracking.

Can be implemented recursively or iteratively using a stack.

Useful for detecting cycles, topological sorting, and finding connected components.

Can be applied to directed/undirected and weighted/unweighted graphs.

Time complexity: O(V + E)

2.2 Breadth-First Search (BFS)

Visits vertices level by level, exploring all neighbors of a vertex before moving to the next level.

Implemented using a queue.

Finds the shortest path in unweighted graphs.

Can be applied to directed/undirected graphs and weighted graphs with equal weights.

Time complexity: O(V + E)

3. Key Concepts

Visited Array: Keeps track of visited vertices to prevent revisiting and infinite loops.

Queue (for BFS): Maintains vertices to explore in order of discovery.

Stack / Recursion (for DFS): Maintains the path being explored before backtracking.

4. Applications of Graph Traversal

- Shortest path finding in unweighted graphs (BFS)
- Cycle detection (DFS)
- Connectivity checks
- Topological sorting (DFS)
- Solving puzzles and games (DFS/BFS)
- Web crawling and network analysis

5. Traversal Strategies Summary

Algorithm	Implementation	Data Structure	Time Complexity	Notes
BFS	Iterative	Queue	O(V + E)	Level-order traversal; shortest path in ur
DFS	Recursive / Iterative	Stack / Recursion	O(V + E)	Depth-first; useful for topological sort an