Graph Traversal Algorithms

Graph traversal algorithms are techniques used to traverse or visit all the nodes or vertices of a graph in a systematic manner. Graphs can be represented in various forms, such as adjacency matrix or adjacency list, and they can be either directed or undirected. There are two primary types of graph traversal algorithms: depth-first search (DFS) and breadth-first search (BFS). These algorithms are used for a variety of applications, including finding paths, detecting cycles, and exploring graphs.

Depth-First Search (DFS):

DFS explores as far as possible along each branch before backtracking. It uses a stack (either an explicit data structure or the call stack in the case of recursion) to keep track of nodes to be visited. DFS has the advantage of consuming less memory compared to BFS

Breadth-First Search (BFS):

BFS explores all the vertices at the present depth before moving on to the vertices at the next depth level. It uses a queue data structure to keep track of nodes to be visited. BFS is particularly useful for finding the shortest path in an unweighted graph.

Application:

Maze Solving:

Graph traversal algorithms can be used to solve mazes. The maze is represented as a graph where each cell is a node, and the edges represent possible moves between adjacent cells. DFS or BFS can be employed to find a path from the start to the exit of the maze.

Social Network Analysis:

In a social network graph, where users are nodes and connections between users are edges, graph traversal can help find connections between users, determine the shortest path between two users, and analyze the overall structure of the network.