Graph traversal algorithm

Graph traversal algorithms are used to explore or visit all the nodes or vertices of a graph. There are two commonly used graph traversal algorithms: Depth-First Search (DFS) and Breadth-First Search (BFS).

Depth-First Search (DFS):

DFS explores the graph by going as deep as possible before backtracking. It starts at a given node and explores as far as possible along each branch before backtracking. Here's the basic idea of the DFS algorithm:

Mark the starting node as visited.

Explore an unvisited adjacent node.

If there are no unvisited adjacent nodes, backtrack to the previous node and repeat the process.

DFS can be implemented using recursion or a stack data structure.

Breadth-First Search (BFS):

BFS explores the graph by visiting all the neighbors of a node before moving on to the next level of neighbors. It starts at a given node and explores all the vertices at the present depth before moving on to vertices at the next depth level. Here's the basic idea of the BFS algorithm:

Mark the starting node as visited and enqueue it.

Dequeue a node and visit all its unvisited neighbors.

Enqueue the unvisited neighbors.

Repeat the process until the queue is empty.

BFS can be implemented using a queue data structure.

Both DFS and BFS have their own advantages and use cases. DFS is often used to search for a particular node or to explore all connected components of a graph. BFS, on the other hand, is useful for finding the shortest path between two nodes or for exploring a graph in a level-by-level manner.

It's worth noting that there are other graph traversal algorithms as well, such as Dijkstra's algorithm for finding the shortest path in weighted graphs and A\* algorithm for heuristic-based search. The choice of the algorithm depends on the specific requirements of the problem at hand.