Artificial Intelligence Lab (BTCOL 706)

Experiment No – 03

Aim: Write a Program to solve any problem using Depth First Search.

Theory:

Depth-First Search (DFS) is an algorithm used for traversing or searching tree or graph data structures. It starts from the root node (or an arbitrary node) and explores as far as possible along each branch before backtracking. In other words, it goes as deep as possible in one direction before exploring other paths. DFS is implemented using either recursion or a stack data structure.

Here are the key characteristics and concepts associated with DFS:

1. **Stack or Recursion**: DFS can be implemented using either an explicit stack data structure or by making use of the program's call stack (recursion). When using recursion, the call stack effectively serves as the stack for tracking nodes to be visited.
2. **Depth-First**: The name "Depth-First Search" comes from the fact that the algorithm explores the deepest nodes in the tree or graph before backtracking. It traverses down the depth of one branch of the tree or graph before exploring neighboring branches.
3. **Backtracking**: When DFS reaches a leaf node or a node with no unvisited neighbors, it backtracks to the nearest unexplored node with unvisited neighbors. This process continues until all nodes are visited.
4. **Visited Nodes**: To avoid revisiting nodes and getting stuck in infinite loops, DFS keeps track of visited nodes. Typically, it maintains a data structure (e.g., a list or set) to record visited nodes.
5. **Applications**: DFS is used in a variety of applications, including finding paths in a maze, topological sorting, cycle detection, and solving problems related to connected components and graph traversal.
6. **Completeness**: In a finite graph or tree, DFS is guaranteed to visit every node if implemented correctly. However, it may not find the shortest path between two nodes.

7. **Memory Usage**: Depending on the implementation, DFS can use less memory compared to breadth-first search (BFS) because it only needs to remember the path from the start node to the current node.
8. **Time Complexity**: The time complexity of DFS is O(V + E), where V is the number of nodes (vertices) and E is the number of edges in the graph. It explores each edge and node at most once.

In summary, Depth-First Search is a fundamental graph traversal algorithm that explores tree and graph structures by going as deep as possible along one branch before backtracking. It is used in various applications and is characterized by its depth-first exploration strategy and use of a stack (or recursion) to keep track of nodes to visit.

Depth First Search Algorithms:

1. Start at a source node (or a starting point in a graph).
2. Mark the source node as visited to prevent revisiting it.
3. Explore an unvisited neighbor of the current node. If there are multiple unvisited neighbors, choose one.
4. If you find an unvisited neighbor, move to that neighbor, mark it as visited, and repeat step 3.
5. If there are no unvisited neighbors, backtrack to the previous node (pop from the stack or return from the recursive call) and continue the exploration.
6. Repeat steps 3-5 until you've visited all reachable nodes or found a specific target node (if you're searching for a particular element).
7. If you're using a stack for iteration, continue popping and exploring nodes until the stack is empty.

Program: Write a following program and take print with output and attached

1. Write a Program in Prolog to solve any problem using Depth First Search.
2. Write a Program in Python to solve any problem using Depth First Search.

Questions:

1. Explain Depth First Search Technique with example.
2. State Advantages and Disadvantages of Depth First Search.
3. Explain Application of Depth First Search Technique

.

(Subject In-charge)

(Prof.S.B.Mehta)