Dabir Aishwarya Sharad
19c0012

Assignment - 1

Aim : Implement depth first search algorithm and breadth first search algorithm. Use an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.

Theory : 1. Breadth first search :

- BFS is the most common search strategy for traversing a tree or graph. This algorithm searches breadthwise in a tree or graph, so it is called breadth first search.

- BFS algorithm starts searching from the root node of the tree and expands all successor node at the tree and expands all successor node at the current level before moving to the node of next level.

- BFS is an example of a general graph search algorithm.

- BFS implemented using FIFO queue data structure.

- Aim of BFS is to traverse the graph as close as possible to root node.

Advantage :

BFS will find the shortest path between the starting point and any other reachable node. A depth first search will not necessarily find the shortest path.

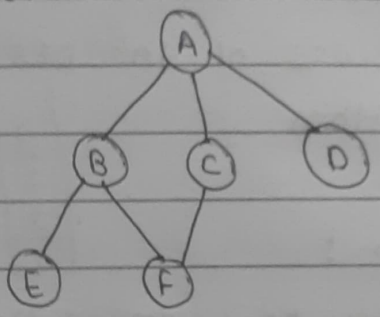
• Disadvantage

A BFS on a binary tree generally requires more memo
-ry than a DFS. If a solution is far away then it
consumes time.

Applications of BFS :

1. Finding the shortest path
2. Checking graph with pettiness
3. Copying cheney's algorithm.

Example of BFS



Start node : A

Goal node : D.

visited [A]

Queue [B, C, D]

visited [A, B]

Queue [C, D, E, F]

visited [A, B, C]

Queue [D, E, F]

visited [A, B, C, D]

Queue [

Goal : D

2. Depth search algorithm

- DFS is recursive algorithm for traversing a tree or graph data structure.

- It is called depth first because it starts from root node and follows each path to its greatest depth node before moving to next path.

- DFS uses a stack data structure for its imple-mentation.

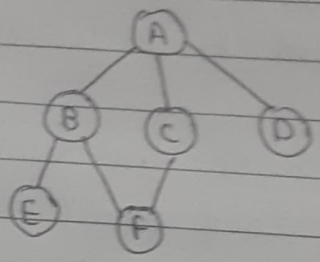
- The process of the DFS algorithm is similar to the BFS algorithm.

Advantage :
The memory requirement is linear WRT nodes. It require less time and space complexity rather than BFS. The solution can be found out without much more search.

Disadvantage :
Not guaranteed that will give you a solution. Cut off depth is smaller so time complexity is more.

Applications of DFS :
1. Finding connected components.
2. Topological sorting
3. Finding bridges of the graph.

• Example of DFS



Start node : A
Goal node : D

visited : [A]
Stack :

B
C
D

visited : [A, B]
stack :

E
F
C
D

visited : [A, B, E]
stack :

F
C
D

visited : [A, B, E, F]
stack :

C
D

visited : [A, B, E, F, C]
stack : [D]

visited : [A, B, E, F, C, D]
stack : Empty.

Goal : D
Path : A → B → E → F → D.

Algorithm :

1. Breadth First Search

Step 1 : Push the root node in the queue
Step 2 : Loop until the queue is empty
step 3 : Remove the node from the queue.
step 4 : If the removed node has unvisited child nodes, mark them as visited and insert the unvisited children in the queue.

2. Depth first search.

Step 1 : Push the root node in the stack
Step 2 : Loop until stack is empty
Step 3 : Peek the node of the stack.
step 4 : If the node has unvisited child nodes ge the unvisited child node, mark it as trave -ed and push it on stack.
Step 5 : If the node does not have any unvisite child nodes, pop the node from the stack

Conclusion Implementation of BFS and DFS is done.