A Mini-Project Report on

BREADTH FIRST TRAVERSAL

Submitted in III Semester

Bachelor of Engineering
In
Computer Science and Engineering

Submitted by

SHETTY PREETHIK LAXMAN	**1DS17CS081**
SATHVIK TN	**1DS17CS101**
SHRAVANAKUMAR BAJANTRI	**1DS17CS106**
SHREESHA MG	**1DS17CS107**

Under the guidance of
SAHANA DAMALE,
Asst. Professor,
Dept. of CSE, DSCE

2018-2019

Department of Computer Science and Engineering,
DAYANANDA SAGAR COLLEGE OF ENGINEERING
BANGALORE – 560078

BFS OF A BINARY TREE

SYNOPSIS:

The project mainly deals with breadth-first traversing of the nodes of a tree. This is used to searching of a tree or graph data structures.

Breadth first search (BFS) is a traversing algorithm where you start traversing from a selected node (source or starting node) and traverse the tree layer-wise, thus exploring the neighbor nodes (nodes which are directly connected to source node). We move towards the next-level neighbor nodes.

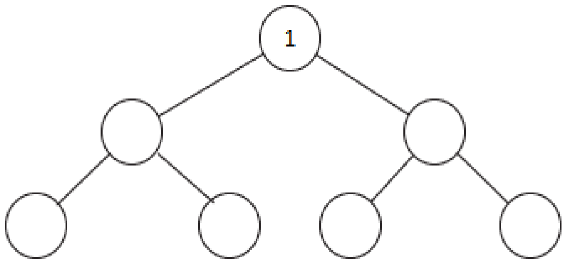
Breadth first search (BFS) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key'), and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level.

TABLE OF CONTENTS

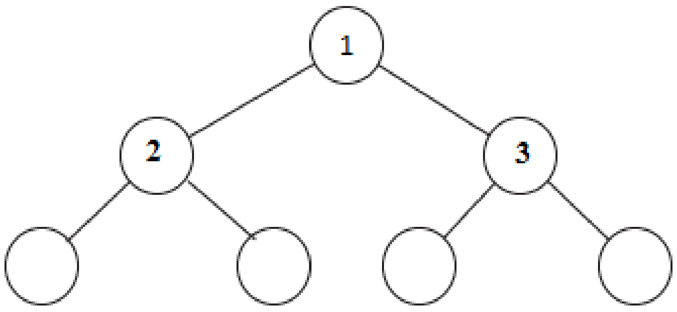
INTRODUCTION:

Breadth-first search (BFS) is a method for exploring a tree or graph. In a BFS, you first explore all the nodes one step away, then all the nodes two steps away, etc. Breadth-first search is like throwing a stone in the center of a pond. The nodes you explore "ripple out" from the starting point.

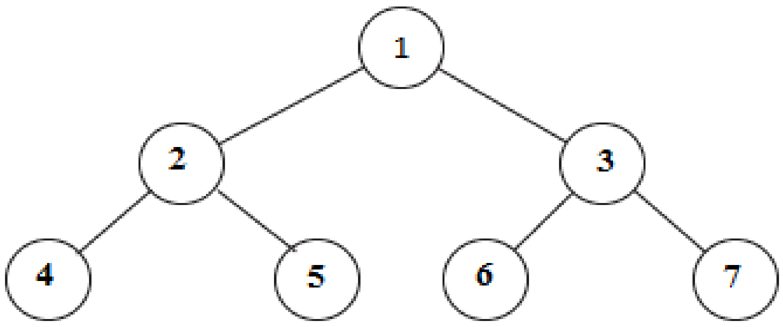
Here's a how a BFS would traverse this tree, starting with the root:



We'd visit all the immediate children (all the nodes that are one step away from our starting node):



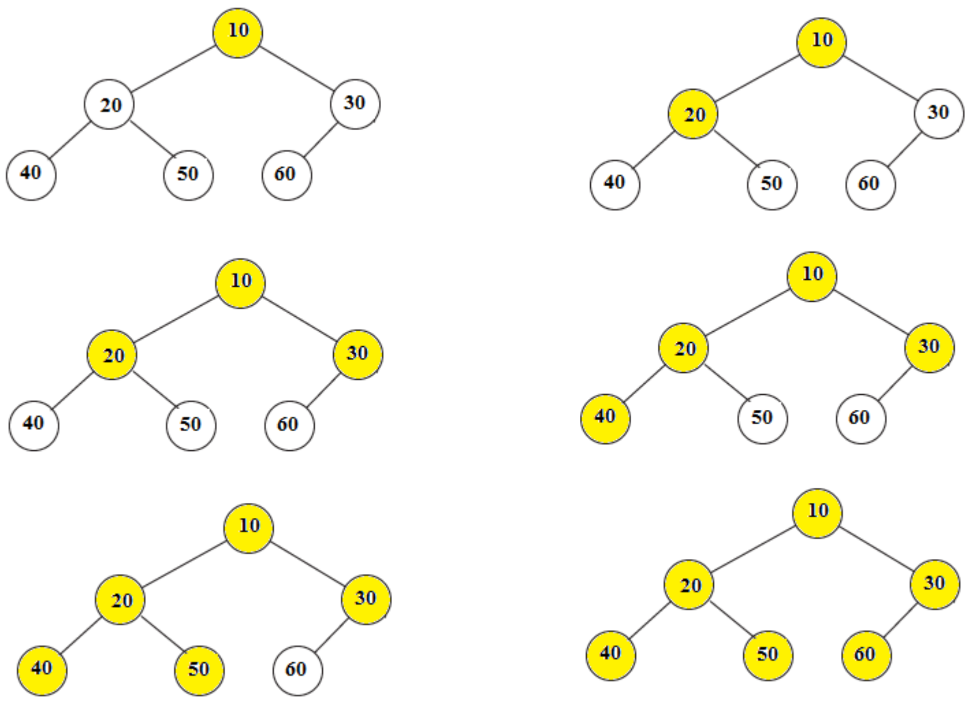
Then we'd move on to all those node's children (all the nodes that are two steps away from our starting node):



And so on, until we reach the end. Breadth-first search is often compared with depth-first search.

IMPLEMENTATION:

The traversing will start from the source node(root) and push(enqueue) root to queue.



According to the fig, implementation is explained here.

Step 1:

- Root 10 will be popped from the queue and printed.
- Children of root i.e. 20 and 30 will be traversed and enqueued to queue.
- 10 is marked as ‘visited’.

Step 2:

- 20 is popped from the queue.
- Children of 20 i.e. 40 and 50 are traversed and enqueued to queue.
- 20 is marked as ‘visited’.

Step 3:

- 30 is popped from the queue.
- Children of 30 i.e. 60 is traversed.
- 30 is marked as ‘visited’.

Step 4:

- 40 is popped from the queue.
- Since there are no children, there will be no enqueue.
- 40 is marked as ‘visited’.

Step 5:

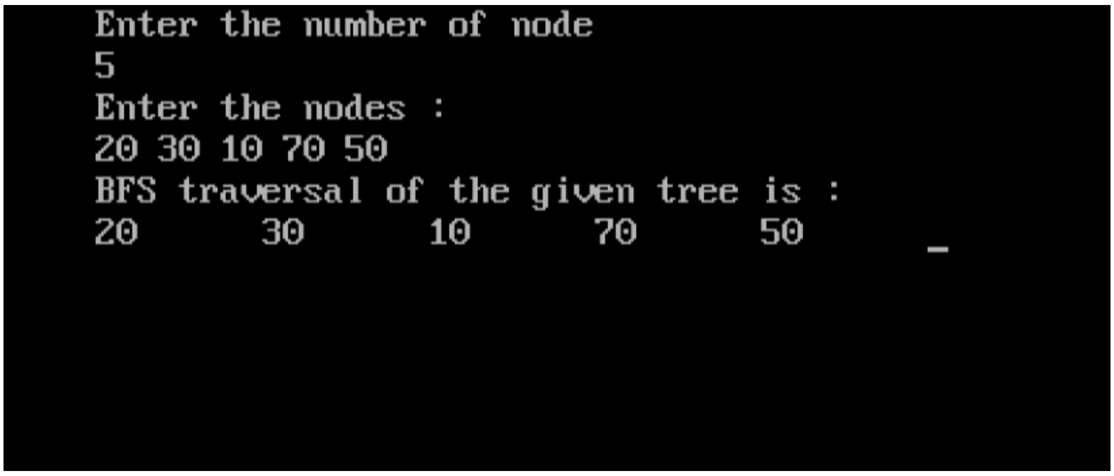
- 50 will be popped from the queue.
- Since there are no children, there will be no enqueue.
- 50 is marked as 'visited'

Step 6:

- 60 is popped from the queue
- Since there are no children, there will be no enqueue.
- 60 is marked as 'visited'

The queue is empty and it comes out of the loop. All the nodes have been traversed by using BFS.

RESULT:



```
Enter the number of node
5
Enter the nodes :
20 30 10 70 50
BFS traversal of the given tree is :
20      30      10      70      50      _
```

ANALYSIS/CONCLUSIONS:

This project deals with BFS where you should start traversing from a selected node (source or starting node) and traverse the tree layer-wise thus exploring the neighbor nodes (nodes which are directly connected to source node). We move towards the next-level neighbor node.

REFERENCES:

1 www.hackerearth.com

2 www.interviewcake.com

3 www.wikipedia.org