International Journal of Engineering & Technology, 7 (4) (2018) 3354-3356

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.12786
Research paper

A new approach for converse binary tree traversals

K. Pushpa Rani [1] *, G. Roja [1], Ch. Sabitha [2], B. Dhana Lakshmi [3], S. Sreeja [4]

[1] *Department of Computer Science and Engineering, MLR Institute of Technology, Hyderabad-500043,India*
[2] *Department of Computer Science and Engineering ,Bardhhaman College of Engineering, Hyderabad-501218,India*
[3] *Department of Computer Science and Engineering, IARE College of Engineering, Hyderabad,-500043, India*
[4] *Department of Computer Science and Engineering, St. Martin's College of Engineering, Hyderabad-500014,India*
**Corresponding author E-mail:*

Abstract

Traversing through the binary tree is the method the way toward going to every node in a predetermined order. There are two ways to deal B with traverse a tree: traversal and converse traversal. Most approaches demonstrate this method using recursive procedures only. Our review paper concentrates on non recursive converse tree traversal .Converse Tree traversal reduces the time complexity if the left sub tree is not present in the binary tree. Converse Tree traversal is similar to tree traversal but in the Converse Right sub tree then left sub tree. The new approach was found to compare tree traversal with converse tree traversal algorithms.

Keywords: *Binary Tree, Pre; In; Post Order; Converse Traversal.*

1. Introduction

Trees are very important in environment without tree the life of human being is difficult. Every tree contains one special part called root and it is having so many branches .In computer science Trees plays a major role to maintain file directory compiler design, Educational organization, and to evaluate arithmetic expressions. In tree special node called the root, the root has no parent, and the branches are called paths or edges. There are different types of trees in computer science. One of the most important trees is Binary Tree. In Binary Tree every internal node has at most two children we can perform some operations on Binary Tree. One operation is Traversal means visiting each and every node exactly once. There are [3] Traversal techniques Preorder, In-order, Post-order. In preorder root node, left sub-tree is visited first, then right sub-tree is visited. In in-order Traversal Left Sub-tree, root node, then right sub-tree. We propose new approach called converse Tree traversal and it follows reverse process of tree traversal. In this, Right sub tree then left sub-tree. Converse tree traversal is a recursive process that means visiting the sub-tree either left or right repeated until no node is found. Most textbooks are written general tree traversal mechanism but there is no complete implementation but we propose new approach along with the C Code implementation. The proposed algorithm is easy to understand and it uses stack data structure.

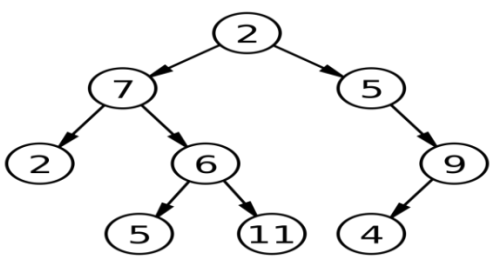


Fig. 1: Binary Tree.

2. Algorithm

Recursive Algorithm For binary tree traversal
Traversal means process each node in some predefined order. Every node in the tree has two parts Left Sub tree and Right Sub tree. Sub tree is a part of the tree either left or right. Visiting a node is done in three ways [3]

1) Pre-Order:
1) Root node is visited first
2) Visit the left sub tree recursively in preorder fashion.
3) Visit the right sub tree recursively in preorder fashion.
2) In-Order:
1) Visit the left sub tree recursively in preorder fashion.
2) Root node is visited
3) Visit the right sub tree recursively in preorder fashion.
3) Post-Order:
1) Visit the left sub tree recursively in preorder fashion.
2) Visit the right sub tree recursively in preorder fashion.
3) Root node is visited

In-Order Traversal:
Algorithm:
Input: Binary Tree with left and right sub tree
Output: In-order traversal of a binary tree
Procedure:
If the root is not null

1) Visit the left sub tree recursively in In order fashion
2) Visit the root node
3) Visit the right sub tree recursively in In order

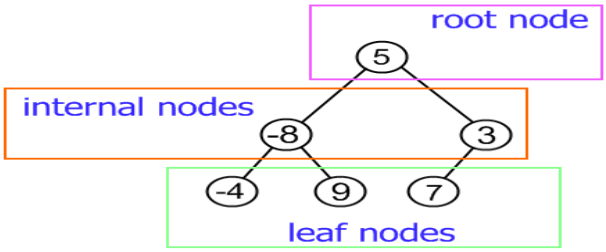


Fig. 2: In-Order Traversal.

3. Non recursive converse in-order traversal algorithm

The normal implementation differs from converse implementation in step1 and step 3.Each traversal requires different strategy. According to weiss [1] [2] inserting a node into the stack 1st, 2nd and 3rd time is depends upon the type of traversal. He uses variable count which is changed when the node is deleted from the stack.
Here we propose a new method for converse in-order traversal without variable count. We use stack data structure
Algorithm:

1) Initialize stack is empty S
2) When node data is not NULL
a) Insert node data into the stack
b) Assign node->right to node.
c) Close the Loop
3) Display all nodes data in in-order

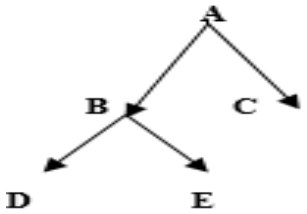
3.1. When node data is not NULL

a) Delete node data from the stack
b) Print the data in in-order fashion
c) Assign node->left to node.

3.2 When node data is not NULL

a) Print the data in in-order fashion
b) Assign node->right to node.
c) Close the Loop
4) Close the Loop

4. Implementation



1) Create an empty stack S=NULL
2) Set new node as address of root: new node->A
3) Pushes the new node and set new node =new node->right until new node is NULL

New node->A
Push A: Stack S->A
New node->C
Push D: Stack S->A,C
New node->NULL

4) If the new node is NULL then pop

Pop C
Stack S: A
Stack S: NULL

5) Set new node is new node->left until new node is NULL

New node->B.
Push B: Stack S->B
New node->D
Push D: Stack S->B, D
New node->NULL
Pop D
Stack S:B
Pop B
Stack S: NULL
New node->NULL
New node->E
Push E: Stack S->E
New node->NULL
Pop E
Stack is empty then stop
OUTPUT: C A D B E
Algorithm for Converse Pre-order And Post-order Traversal
Algorithm: CONVERSE PRE ORDER

1) Initialize stack is empty S
2) Insert root node data into the stack
3) Display the data in Pre-order
4) When node data is not NULL
a) Delete node data from the stack
b) Display the node data in pre-order.
5) If the node right is not NULL
a) Insert node->right into the stack
6) If the node left is not NULL
a) Insert node->left into the stack
7) Close Loop

Algorithm: Converse Post-Order
Algorithm: CONVERSE PRE-ORDER

1) Initialize stack is empty S
2) If the node right is not NULL

a. Insert node->right into the stack

3) If the node left is not NULL

a. Insert node->left into the stack

4) When node is not NULL
a) Delete node data from the stack
b) print the data in preorder fashion
5) Insert root node into the stack
6) Display the node data in Pre-order
7) Close Loop.

5. Results

Enter data (1 for no data):1
Enter left child of 1:
Enter data (1 for no data):2
Enter left child of 2:
Enter data (1 for no data):4
Enter left child of 4:
Enter data (1 for no data):-1
Enter right child of 4:
Enter data (1 for no data):-1
Enter right child of 2:
Enter data (1 for no data): five
Enter left child of 5:
Enter data (1 for no data):-1
Enter right child of 5:
Enter data (1 for no data):-1
Enter right child of 1:
Enter data (1 for no data):3
Enter left child of 3:
Enter data (1 for no data):-1
Enter right child of 3:
Enter data (-1 for no data):-1
The preorder traversal of tree is:
1
2
4
5
3
The postorder traversal of tree is:
4

5
2
3
1
The in order traversal of tree is:
4
2
5
1
3
TThe converse preorder traversal of tree is:
1
3
2
4
5
The Converse postorder traversal of tree is:
3
4
5
2
1
The Inorder traversal of tree is:
3
1
4
2
5[Inferior 1

6. Conclusion

In this paper we propose new approach called converse tree traversal in non recursive fashion and it is compared with tree traversal in binary tree. This approach is useful when the left sub tree is empty are less number of children it is easy to understand compared to the previous methods. Instead of pushing the nodes into the stack once, twice, thrice [1-2] and this method reduces time complexity.

References

[1] Weiss Data structures and Problem solving using Java 2 Edition.
[2] D.E.Knuth The art of computing and Problem solving.
[3] https://ieeexplore.ieee.org/abstract/document/1301900/?section=abstract.