145. Binary Tree Postorder Transversal (/problems/binary-tree-postorder-traversal/)

Oct. 12, 2018 | 17.8K views

Average Rating: 3.65 (17 votes)

Given a binary tree, return the *postorder* traversal of its nodes' values.

Example:

```
Input: [1,null,2,3]
   1
    \
     2
    /
   3

Output: [3,2,1]
```

Follow up: Recursive solution is trivial, could you do it iteratively?

Solution

How to transverse the tree

There are two general strategies to transverse a tree:

- *Breadth First Search* (`BFS`)

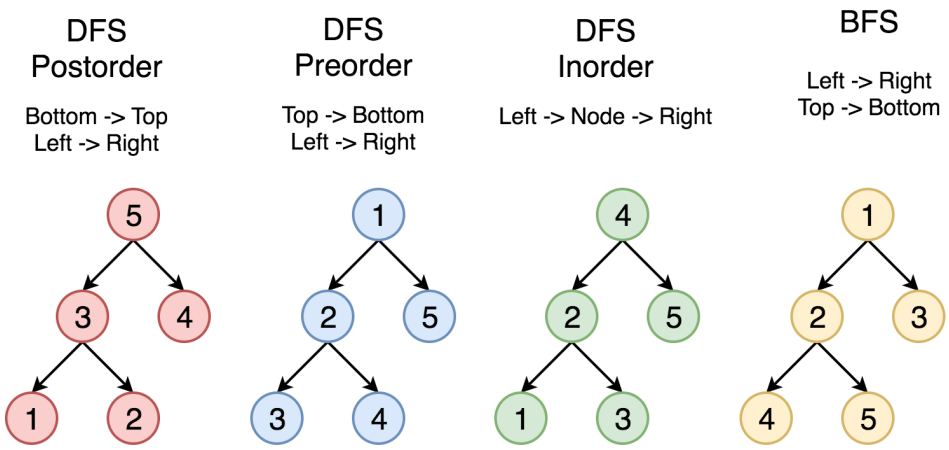
  We scan through the tree level by level, following the order of height, from top to bottom. The nodes on higher level would be visited before the ones with lower levels.

- *Depth First Search* (`DFS`)

  In this strategy, we adopt the `depth` as the priority, so that one would start from a root and reach all the way down to certain leaf, and then back to root to reach another branch.

  The DFS strategy can further be distinguished as `preorder`, `inorder`, and `postorder` depending on the relative order among the root node, left node and right node.

On the following figure the nodes are numerated in the order you visit them, please follow `1-2-3-4-5` to compare different strategies.



Here the problem is to implement postorder transversal using iterations.

Approach 1: Iterations

Algorithm

First of all, here is the definition of the `TreeNode` which we would use in the following implementation.

Java | Python — Copy

```
class TreeNode(object):
    """ Definition of a binary tree node."""
    def __init__(self, x):
        self.val = x
        self.left = None
        self.right = None
```

Let's start from the root and then at each iteration pop the current node out of the stack and push its

child nodes. In the implemented strategy we push nodes into stack following the order `Top->Bottom` and `Left->Right`. Since DFS postorder transversal is `Bottom->Top` and `Left->Right` the output list should be reverted after the end of loop.

Java | Python

Copy

```
class Solution(object):
    def postorderTraversal(self, root):
        """
        :type root: TreeNode
        :rtype: List[int]
        """
        if root is None:
            return []

        stack, output = [root, ], []
        while stack:
            root = stack.pop()
            output.append(root.val)
            if root.left is not None:
                stack.append(root.left)
            if root.right is not None:
                stack.append(root.right)

        return output[::-1]
```

Complexity Analysis

- Time complexity : we visit each node exactly once, thus the time complexity is $\mathcal{O}(N)$, where N is the number of nodes, *i.e.* the size of tree.
- Space complexity : depending on the tree structure, we could keep up to the entire tree, therefore, the space complexity is $\mathcal{O}(N)$.

Analysis written by @liaison (https://leetcode.com/liaison/) and @andvary (https://leetcode.com/andvary/)

Rate this article:

Previous (/articles/maximum-sub-circular-subarray/)

Next (/articles/minimum-add-to-make-parentheses-valid/)

Comments: 6

Sort By

Type comment here... (Markdown is supported)

Articles > 145. Binary Tree Postorder Transversal

Preview Post

(/jianchao-li)

jianchao-li (jianchao-li) ★ 10497 March 9, 2019 1:46 AM

Actually this solution does not visit the nodes in the post order but just tweaks the order of the output.

24 Share Reply

SHOW 3 REPLIES

(/liuyubobobo)

liuyubobobo (liuyubobobo) ★ 163 October 13, 2018 1:34 AM

There are so many ways to complete this classic problem. I offered nine solutions on my Leetcode repo in both C++ (https://github.com/liuyubobobo/Play-Leetcode/tree/master/0145-Binary-Tree-Postorder-Traversal/cpp-0145) and Java (https://github.com/liuyubobobo/Play-Leetcode/tree/master/0145-Binary-Tree-Postorder-Traversal/java-0145/src). :-)

6 Share Reply

SHOW 1 REPLY

(/moyiyiyiyi)

moyiyiyiyi (moyiyiyiyi) ★ 3 June 4, 2019 5:38 PM

？？Why is this listed as hard lol

2 Share Reply

SHOW 1 REPLY

(/ngoc_lam)

ngoc_lam (ngoc_lam) ★ 27 October 12, 2018 7:56 PM

@liaison (https://leetcode.com/liaison) you should add moris traversal for O(1) space :)

2 Share Reply

SHOW 1 REPLY

(/frozenleetcode)

frozenleetcode (frozenleetcode) ★ 6 January 19, 2019 1:13 AM

I wonder if `addFirst()` all the time will lead to a time complexity of `O(N^2)`, cause it will push all the elements back to make space for the new added element

1 Share Reply

SHOW 1 REPLY

(/etherwei)

EtherWei (etherwei) ★ 31 September 28, 2019 11:44 AM

Why this is a hard problem?

0 Share Reply