CS213/293 Data Structure and Algorithms 2023

Lecture 7: Tree walks

Instructor: Ashutosh Gupta

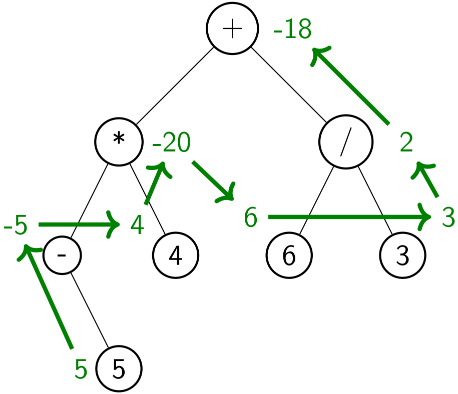
IITB India

Compile date: 2023-08-20

Application : Evaluating an expression

Example 7.1

If we want to evaluate an expression represented as a binary tree, we need to visit each node and evaluate the expression in a certain order.



In green, we have evaluated the value of the node. The path indicates the order of evaluation.

Topic 7.1

Tree walks

Tree walks

Visiting nodes of a tree in a certain order are called tree walks.

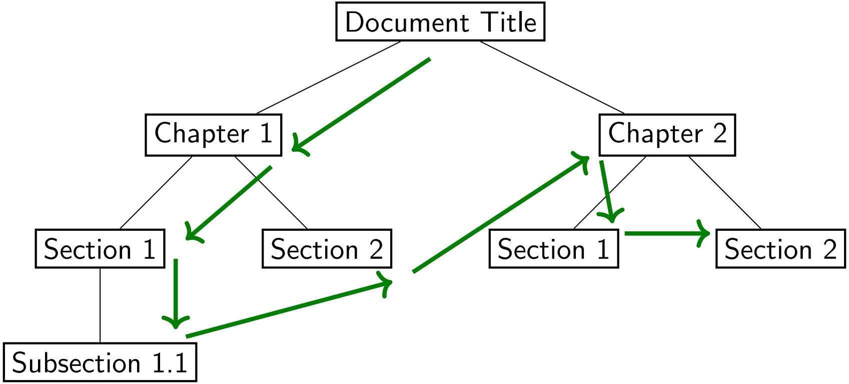
There are two kinds of walks for trees.

- preorder: visit parent first
- postorder: visit children first

Example: preorder

Example 7.2

Let a document be stored as a tree. We read the document in preorder.



Preorder/Postorder walk

Algorithm 7.1: PreOrderWalk(n)

```
visit(n);
for $n' \in children(n)$ do
    PreOrderWalk(n');
```

Algorithm 7.2: PostOrderWalk(n)

```
for $n' \in children(n)$ do
    PostOrderWalk(n');
visit(n);
```

The first example of expression evaluation is postorder walk.

Commentary: visit(v) is some action taken during the walk.

Walking on ordered tree

How do we walk on an ordered tree?

For an ordered tree, we may visit children in the given order among siblings.

We may have choices to change the order of visits among ordered siblings.

Commentary: Our description of the algorithm works for both ordered and unordered trees. Our algorithm does not specify the order of visits of siblings for unordered trees. Please pay attention to the subtle differences among trees, ordered trees, and binary trees.

Topic 7.2

Walking binary trees

Preorder/Postorder walk over binary trees

We have more structure in binary trees. Let us write the algorithm for walks again.

Algorithm 7.3: PreOrderWalk(n)

```
if $n == Null$ then
    return
visit(n);
PreOrderWalk(left(n));
PreOrderWalk(right(n));
```

Algorithm 7.4: PostOrderWalk(n)

```
if $n == Null$ then
    return
PostOrderWalk(left(n));
PostOrderWalk(right(n));
visit(n);
```

Inorder walk of binary trees

Definition 7.1

In an inorder walk of a binary tree, we visit the node after visiting the left subtree and before visiting right subtree.

Algorithm 7.5: InOrderWalk(n)

```
if $n == Null$ then
    return
InOrderWalk(left(n));
visit(n);
InOrderWalk(right(n));
```

Exercise 7.1

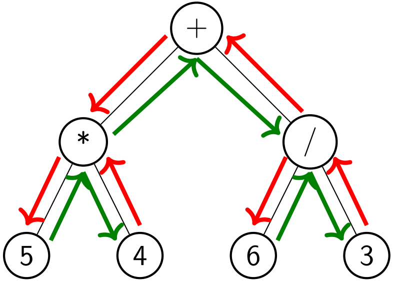
Given a complete binary tree with 7 nodes, label the nodes so that the preorder, inorder, and postorder traversals produce the sequence 1,2,...,7.

Application : Printing an expression

To print an expression (without unary minus), we need to visit the nodes in inorder.

Algorithm 7.6: PrintExpression(n)

```
1 if n is leaf then
2     print(label(n));
3     return
4 print("(");
5 PrintExpression(left(n));
6 print(label(n));
7 PrintExpression(right(n));
8 print(")");
```



((5 * 4) + (6 / 3))

Exercise 7.2

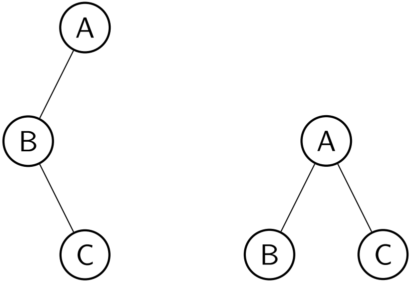
a. Modify the above algorithm to support unary minus.

*b. What will happen if "**if**" at line 1 is replaced by "**if** $n == NULL$ **then return**"?*

Commentary: The order of the walk is the pattern of recursive calls and actions on nodes. An application may need mixed action pattern. In the above printing example, we need to print parentheses before and after making recursive calls. The parentheses are printed pre/post-order. All three walks are present in the above algorithm.

Many trees have the same walks

The following two ordered trees have the same preorder walks.



Commentary: Answer:

For postorder:

```
            A
           /
    A     C
   / \     \
  B   C     B
```

For inorder:

```
    A       B
   / \     / \
  B   C   D   A
 / \         / \
D   E       E   C
```

For postorder and preorder:

```
  A   A
 /     \
C       C
 \     /
  B   B
```

Exercise 7.3

a. Give two binary trees that have the same postorder walks.

b. Give two binary trees that have the same inorder walks.

c. Give two binary trees that have the same postorder and preorder walks.

Topic 7.3

Problems

The uniqueness of walks if two walks are the same.

Exercise 7.4

Give an algorithm for reconstructing a binary tree if we have the preorder and inorder walks.

Exercise 7.5

Let us suppose all internal nodes of a binary tree have two children. Give an algorithm for reconstructing the binary tree if we have the preorder and postorder walks.

Exercise: previous print

Exercise 7.6

For a given binary tree, let $prevPrint(T, a)$ *give the node* n' *such that* $label(n')$ *will appear just before* $label(n)$ *in the inorder printing of* T*. Give a program to compute prevloc.*

Reconstructing tree from preorder walks

Exercise 7.7

Let us suppose we can calculate the number of children of a node by looking at the label of a node of a binary tree, e.g., arithmetic expressions. Give an algorithm for reconstructing the binary tree if we have the preorder walk.

Exercise: level-order walk

Exercise 7.8

Give an algorithm for walking a tree such that nodes are visited in the order of their level. Two nodes at the same level can visit in any order.

Exercise 7.9

Give an algorithm for walking a tree such that nodes are visited in the order of their height.

End of Lecture 7