CSE 595: Advanced Topics in Computer Science
Presentation 5

Zeeshan Shaikh

Department of Computer Science, Stonybrook University

06/23/2021

Topics for today's presentation

- Finding the rank of a given element in a tree.

Problem 5: Finding the rank of a given element in a tree.

What is the rank of an element in a tree?

- The rank of a node holding a key "k" is defined as the total number of nodes in the tree that have a key value strictly less than k.

Problem 5: Rank Function Algorithm

Algorithm 1 Function to find the rank

```
1: Rank() [Start from the parent node of the tree]
2: if node == NULL then
3:     return 0
4: else if node.key > key then
5:     return rank(node.left) [recursive call]
6: else if node.key < key then
7:     return (1 + size(node.left) + rank(node.right))
8: else
9:     return (1 + size(node.left))
10: end if
```

Example of the algorithm

Algorithm 1 Dynamic Programming algorithm

```
1:  Rank() [Start from the parent node of the tree]
2:  if node == NULL then
3:      return 0
4:  else if node.key > key then
5:      return rank(node.left) [recursive call]
6:  else if node.key < key then
7:      return (1 + size(node.left) + rank(node.right))
8:  else
9:      return (1 + size(node.left))
10: end if
```

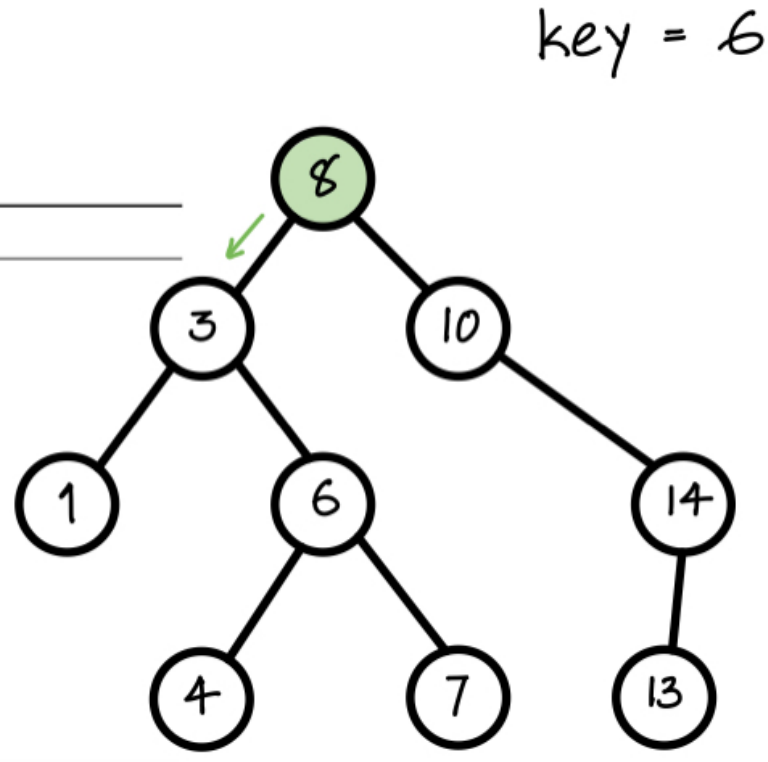


Figure 1: Step 1

Example of the algorithm

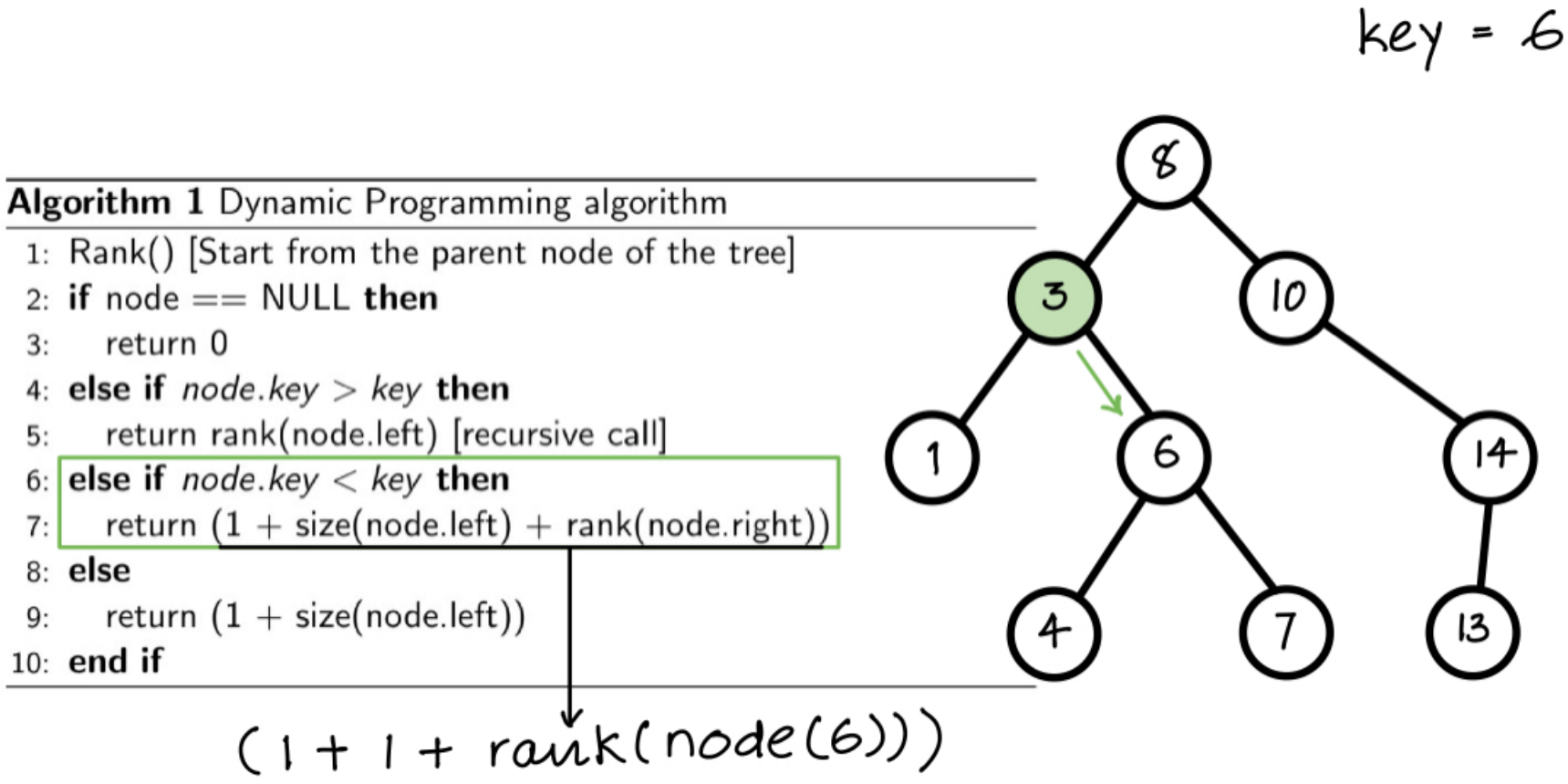
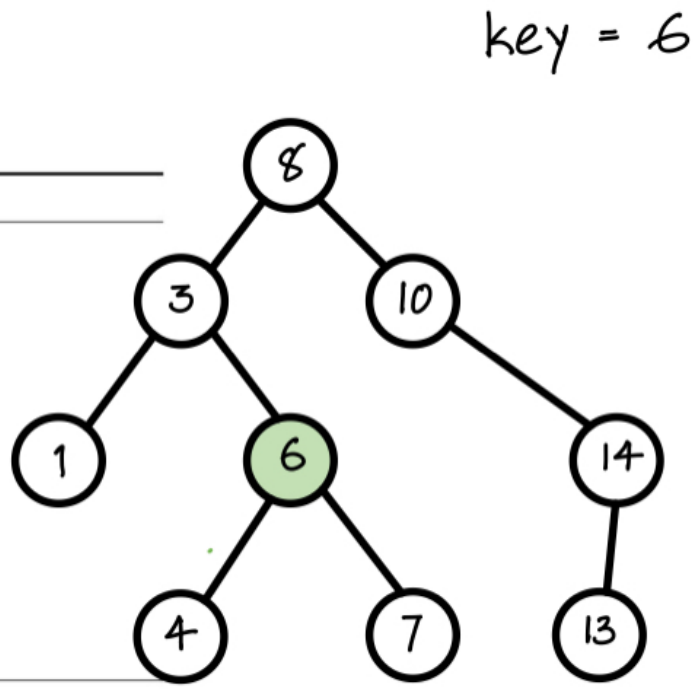


Figure 2: Step 2

Example of the algorithm

Algorithm 1 Dynamic Programming algorithm

```
1:  Rank() [Start from the parent node of the tree]
2:  if node == NULL then
3:      return 0
4:  else if node.key > key then
5:      return rank(node.left) [recursive call]
6:  else if node.key < key then
7:      return (1 + size(node.left) + rank(node.right))
8:  else
9:      return (1 + size(node.left))
10: end if
```



(1 + 1 + rank(node(6))) = (2 + (1 + node(6).left))

= (2 + 1 + 1) = 4 (Rank)

Figure 3: Step 3

Problem 5: Finding the rank of a given element in a tree.

- How do you find the size of the sub-tree used in the previous algorithm?

Problem 5: Rank Function Algorithm

Algorithm 2 Function to find the size of a sub-tree given a node

```
1: Size()
2: if node == NULL then
3:     return 0
4: else
5:     return (1 + size(node.left) + size(node.right))
6: end if
```

Complexity Analysis

Complexity Analysis		
Algorithm	Time	Space
Rank of an element	$K * O(N) * *$	$O(N)$

where K is the number of layers in the tree.