CH-231-A

Algorithms and Data Structures

ADS

Lecture 23

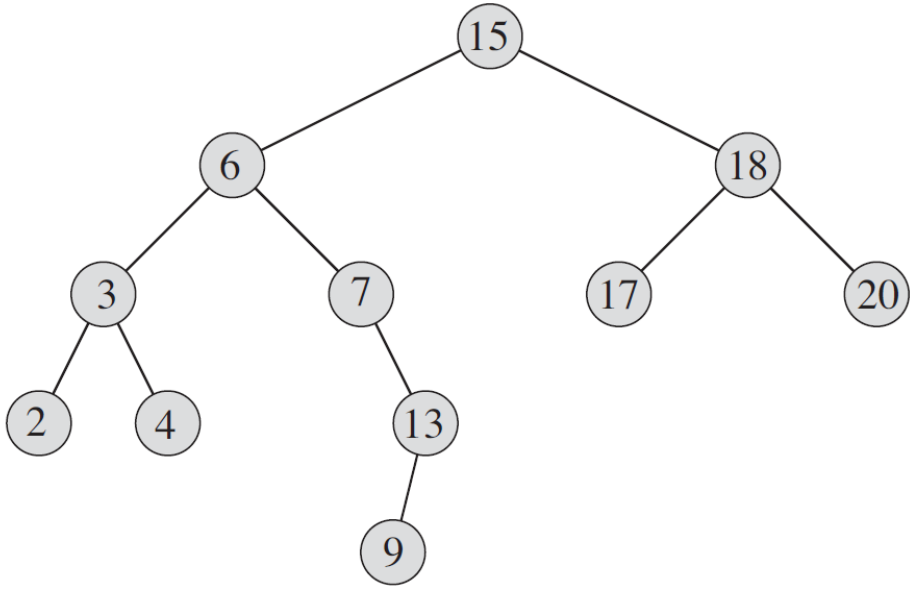
Dr. Kinga Lipskoch

Spring 2022

Query: Finding Minimum / Maximum

```
TREE-MINIMUM(x)
1  while x.left ≠ NIL
2      x = x.left
3  return x
```

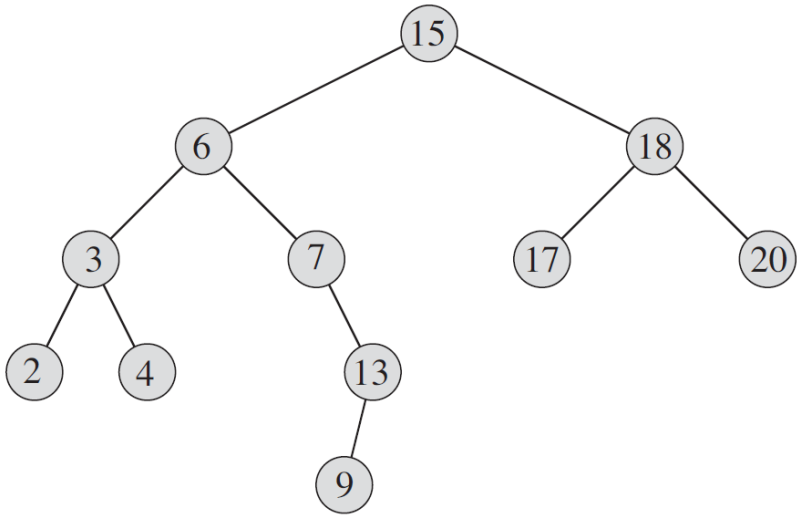
```
TREE-MAXIMUM(x)
1  while x.right ≠ NIL
2      x = x.right
3  return x
```



Time complexity: $O(h)$

Query: Finding Successor (In Order)

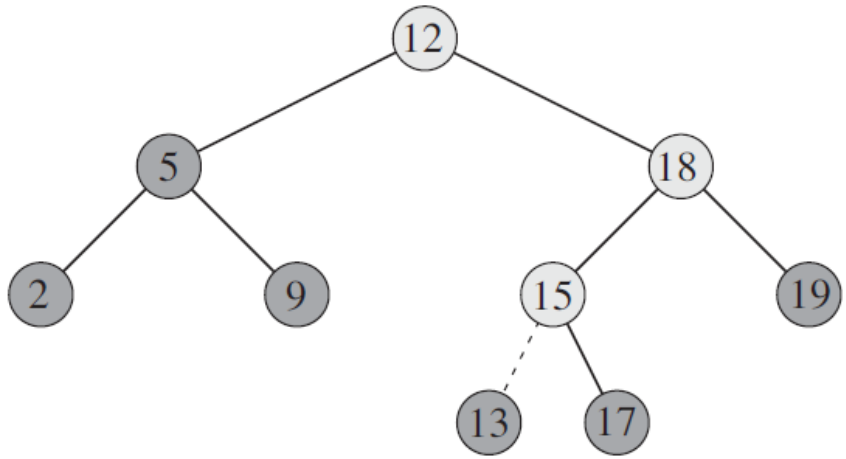
```
TREE-SUCCESSOR(x)
1  if x.right ≠ NIL
2        return TREE-MINIMUM(x.right)
3  y = x.p
4  while y ≠ NIL and x == y.right
5        x = y
6        y = y.p
7  return y
```



Time complexity: $O(h)$

Modify Operation: Insertion (In Order)

```
TREE-INSERT(T, z)
 1   y = NIL
 2   x = T.root
 3   while x ≠ NIL
 4         y = x
 5         if z.key < x.key
 6               x = x.left
 7         else x = x.right
 8   z.p = y
 9   if y == NIL
10         T.root = z
11   elseif z.key < y.key
12         y.left = z
13   else y.right = z
```



Time complexity: $O(h)$

Modify Operation: Transplant

Replaces a subtree rooted at node u with a subtree rooted at node v.

```
TRANSPLANT(T, u, v)
1  if u.p == NIL
2        T.root = v
3  elseif u == u.p.left
4        u.p.left = v
5  else u.p.right = v
6  if v ≠ NIL
7        v.p = u.p
```

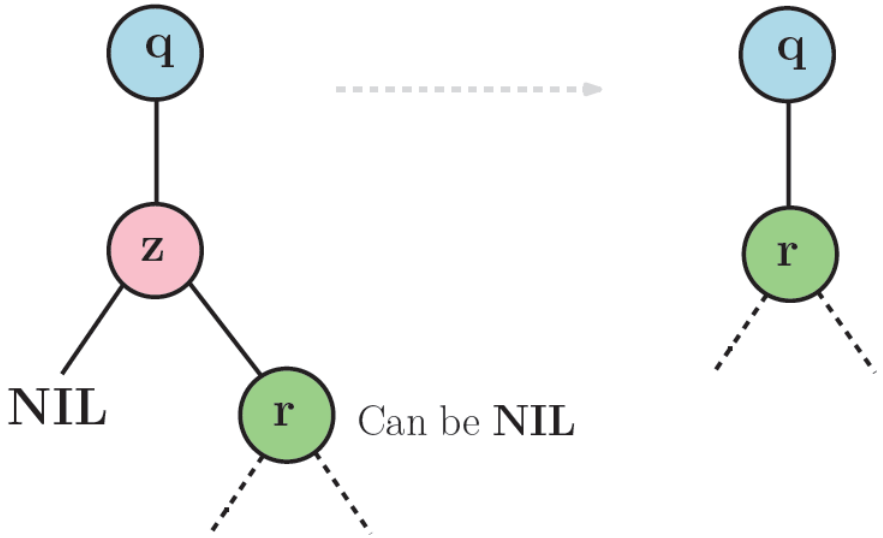
Remarks:

- $u.p$ can be nil.
- v can be nil.
- Time complexity: O(1)

Modify Operation: Deletion (1)

Case 1:

Deleted node z has no or only right child.

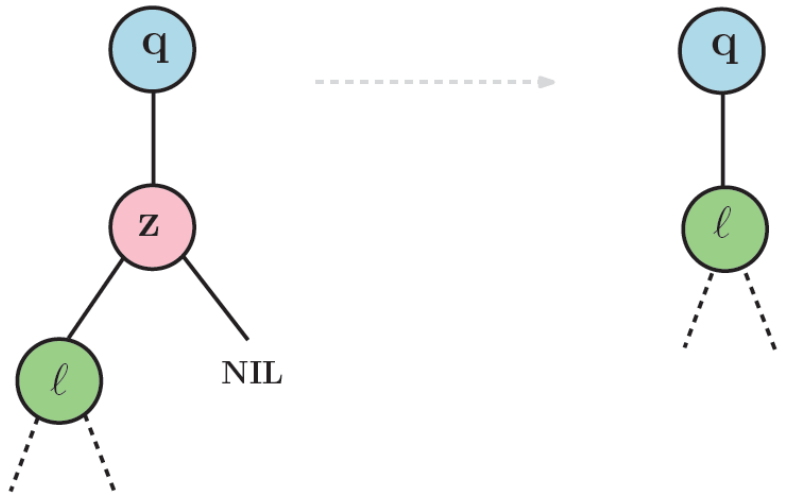



```
1   if z.left == NIL
2       TRANSPLANT(T, z, z.right)
```

Modify Operation: Deletion (2)

Case 2:

Deleted node z has only left child.



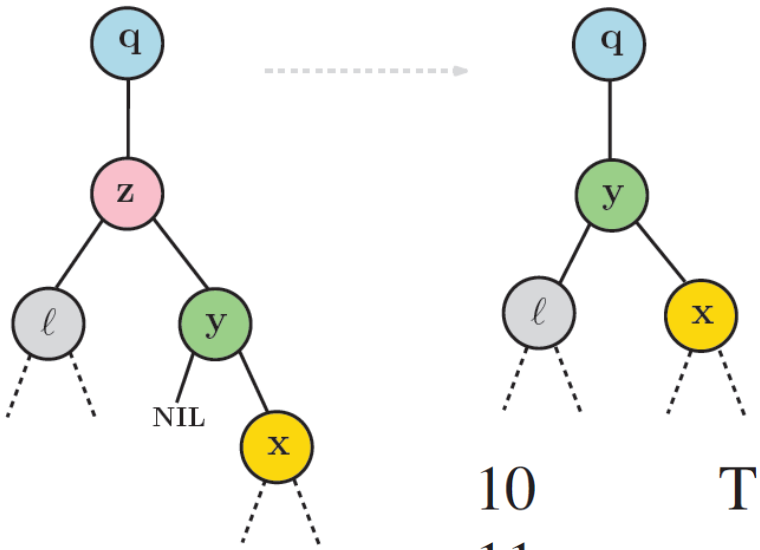

```
3   elseif z.right == NIL
4       TRANSPLANT(T, z, z.left)
```

Remark: For both cases, it does not matter whether z is $q.left$ or $q.right$.

Modify Operation: Deletion (3)

Case 3a:

Deleted node z has both children and $Successor(z) = z.right$.




```
10    TRANSPLANT(T, z, y)
11    y.left = z.left
12    y.left.p = y
```

Modify Operation: Deletion (4)

Case 3b:

Deleted node z has both children and $Successor(z) = y \neq z.right$.