# Design and Analysis of Algorithms

## Lecture-20

Dharmendra Kumar (Associate Professor)
Department of Computer Science and Engineering
United College of Engineering and Research,
Prayagraj

# Deletion operation

Procedure: Suppose we want to delete key k from the B-tree with minimum degree t. Then we use following steps for this purpose:-

1. If the key k is in node x and x is a leaf, then delete the key k from x.

2. If the key k is in node x and x is an internal node, then we do the following:

   2a. If the child y that precedes k in node x has at least t keys, then find the predecessor k’ of k in the subtree rooted at y. Recursively delete k’, and replace k by k’ in x.

## Deletion operation

2b. If y has fewer than t keys, then, symmetrically, examine the child z that follows k in node x. If z has at least t keys, then find the successor k’ of k in the subtree rooted at z. Recursively delete k’ , and replace k by k’ in x.

2c. Otherwise, if both y and z have only t-1 keys, merge k and all of z into y, so that x loses both k and the pointer to z, and y now contains 2t-1 keys. Then free z and recursively delete k from y.

## Deletion operation

3. If the key k is not present in internal node x, then determine the root $x.c_i$ of the appropriate subtree that must contain k, if k is in the tree at all. If $x.c_i$ has only t-1 keys, execute step 3a or 3b as necessary to guarantee that we descend to a node containing at least t keys. Then finish by recursing on the appropriate child of x.

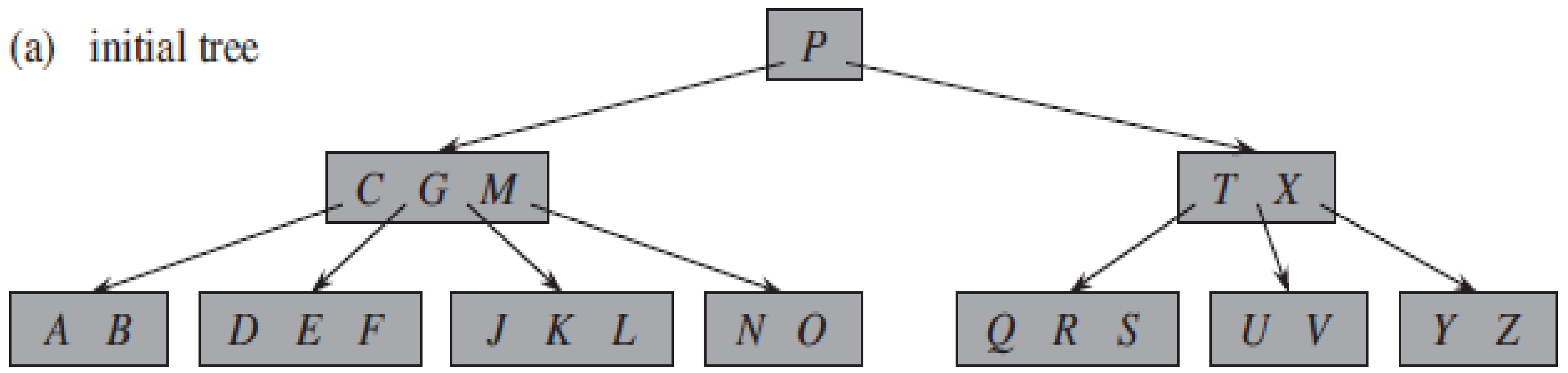
3a. If $x.c_i$ has only t-1 keys but has an immediate sibling with at least t keys, give $x.c_i$ an extra key by moving a key from x down into $x.c_i$, moving a key from $x.c_i$'s immediate left or right sibling up into x, and moving the appropriate child pointer from the sibling into $x.c_i$ .

3b. If $x.c_i$ and both of $x.c_i$'s immediate siblings have t-1 keys, merge $x.c_i$ with one sibling, which involves moving a key from x down into the new merged node to become the median key for that node.

# Deletion operation

Example: Consider the following B-tree with minimum degree t = 3.

(a) initial tree



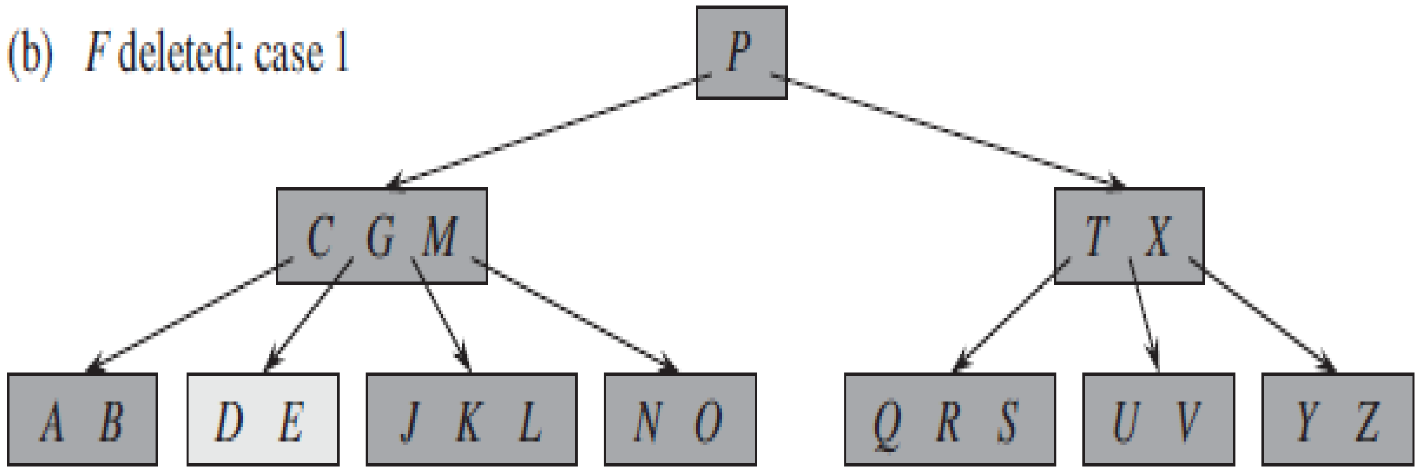
Delete the following elements from this B-tree in-order.

F, M, G, D and B.

# Deletion operation

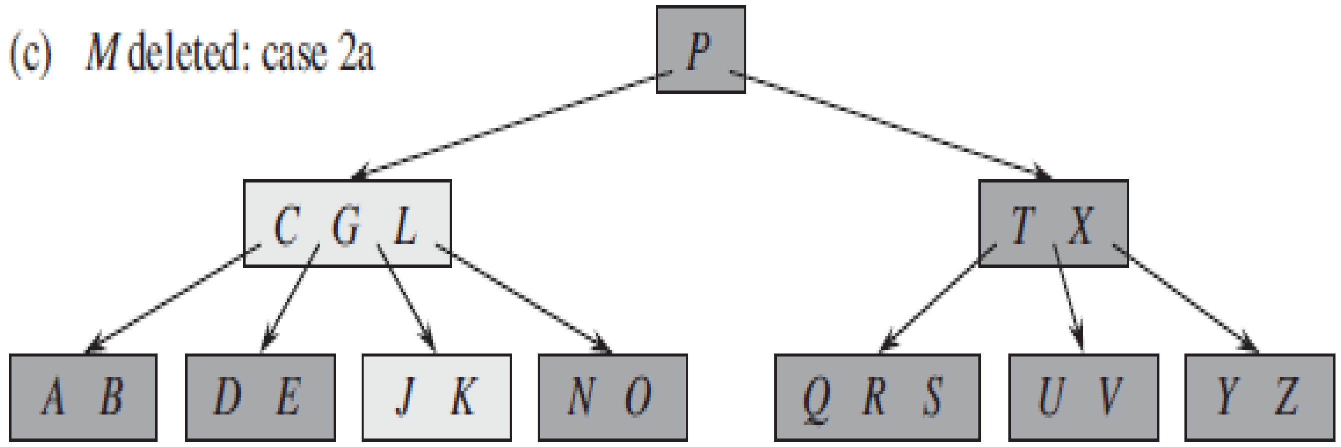
Solution:

Deletion of F



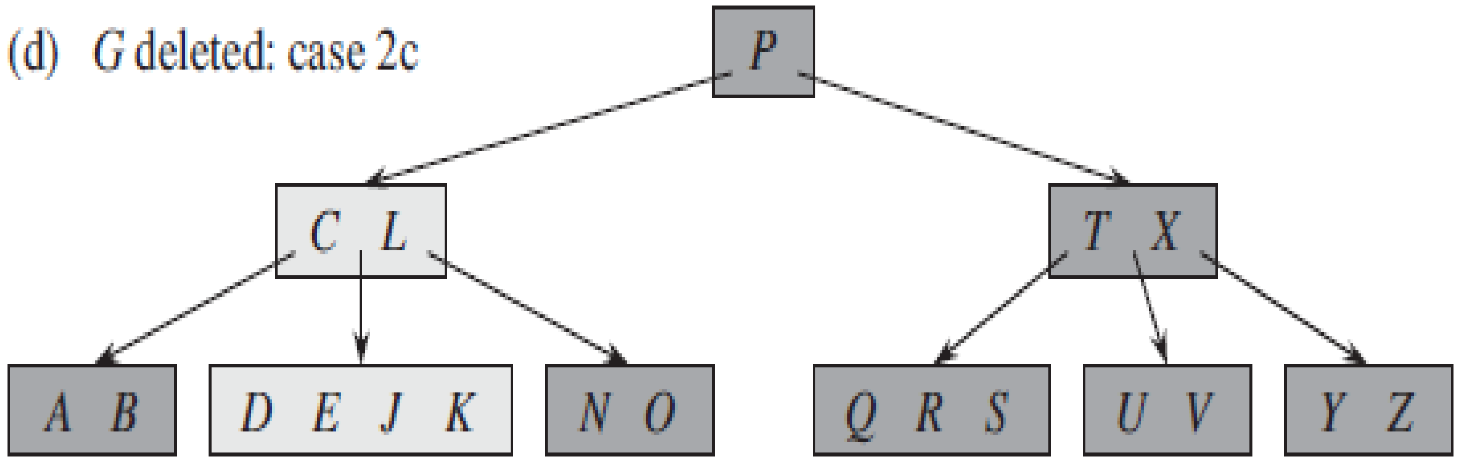
# Deletion operation

Deletion of M



## Deletion operation

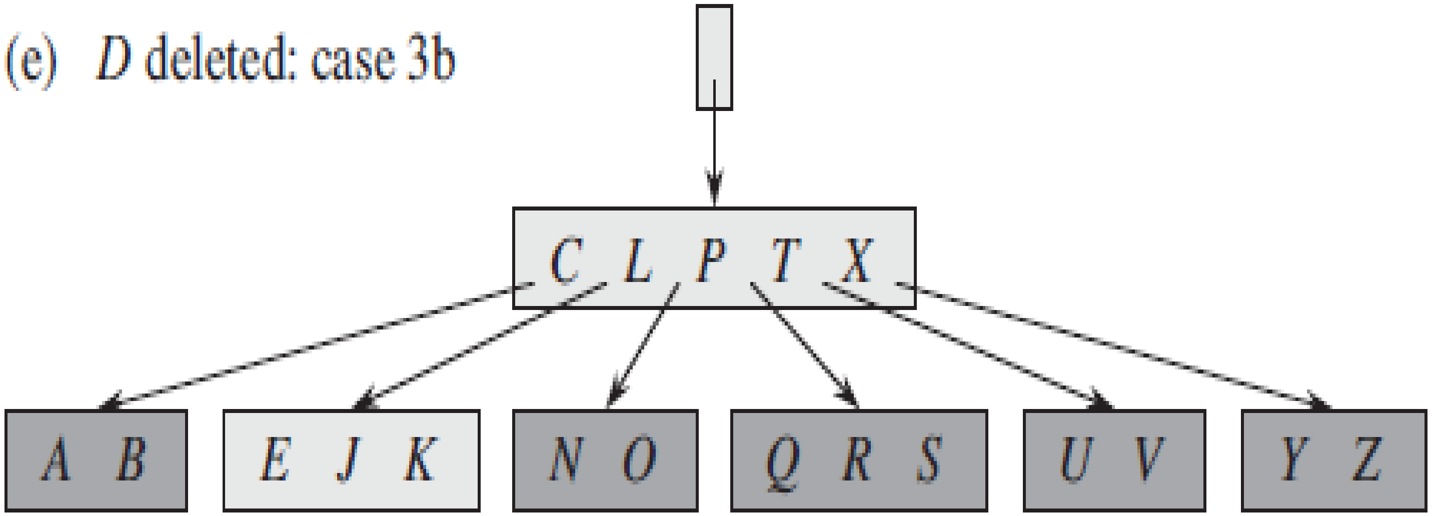
<u>Deletion of G</u>



# Deletion operation

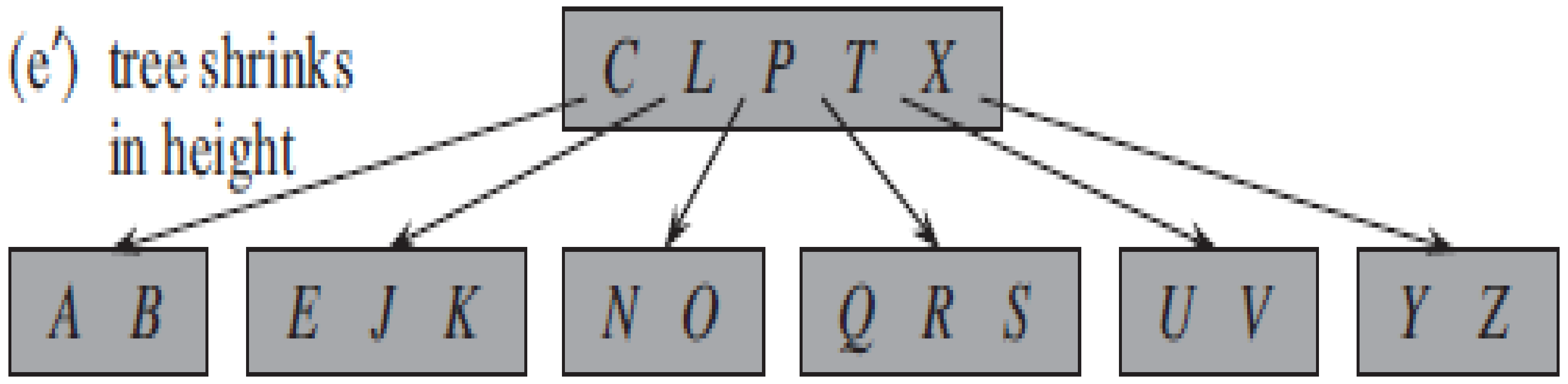
Deletion of D

(e) *D* deleted: case 3b



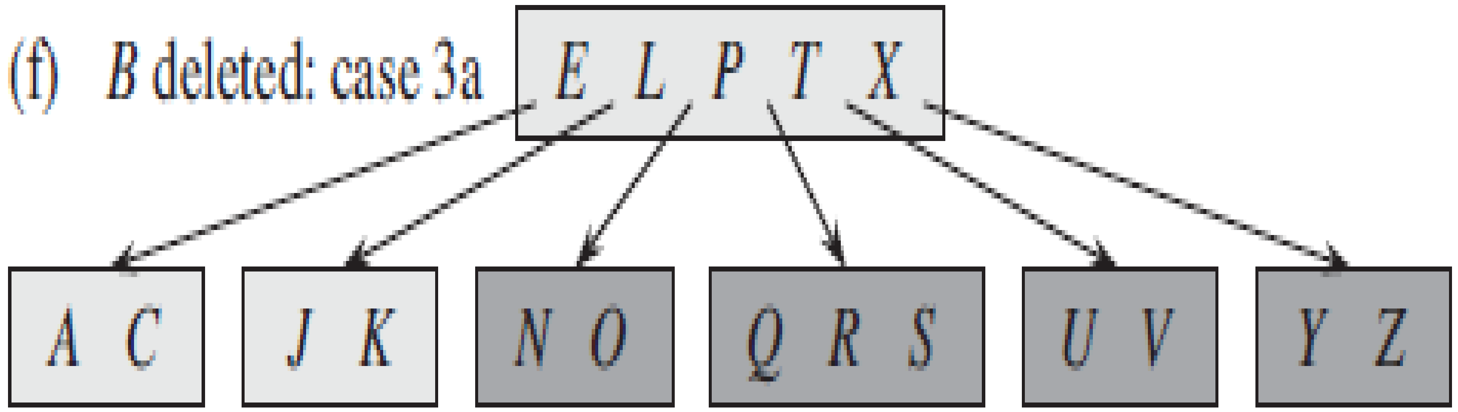
## Deletion operation

Deletion of D



# Deletion operation

Deletion of B



Final B-tree

Time complexity of deletion operation is O(th) i.e.

$O(th) = O(t \log_t n)$

# AKTU Questions

1. Discuss the advantages of using B-Tree. Insert the following Information 86, 23, 91, 4, 67, 18, 32, 54, 46, 96, 45 into an empty B-Tree with degree t = 2 and delete 18, 23 from it.
2. Define a B-Tree of order m. Explain the searching operation in a B-Tree.
3. Using minimum degree ‘t’ as 3, insert following sequence of integers 10, 25, 20, 35, 30, 55, 40, 45, 50, 55, 60, 75, 70, 65, 80, 85 and 90 in an initially empty B-Tree. Give the number of nodes splitting operations that take place.
4. Insert the following information F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, D, Z, E, G, I. Into an empty B-tree with degree t=3.
5. Prove that if n>=1, then for any n-key B-Tree of height h and minimum degree t >=2, h<=log t ((n +1)/2).
6. Insert the following keys in a 2-3-4 B Tree: 40, 35, 22, 90, 12, 45, 58, 78, 67, 60 and then delete key 35 and 22 one after other.

Thank You.