

COL106: Data Structures and Algorithms

Lecture 26

Keerti Choudhary
Department of Computer Science, IIT Delhi

Today's Lecture

Revisit 234 Trees used in Assignment 4

Some Resources:

234 Trees: <https://people.ksp.sk/~kuko/gnarley-trees/234tree.html>

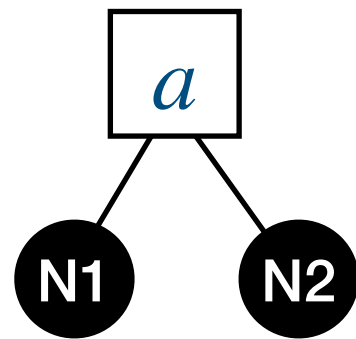
Trees: <https://people.ksp.sk/~kuko/gnarley-trees/Intro.html>

234 Trees

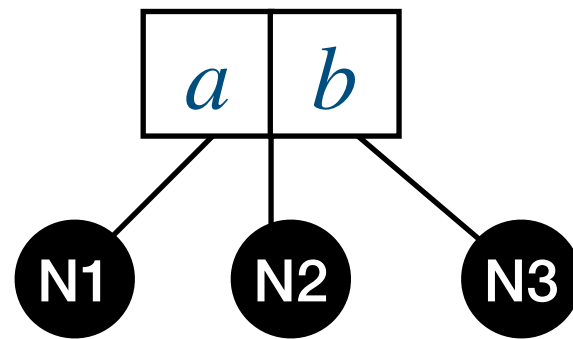
Property 1: Every node is of following type:

- **2-node** : has **one** data element, and **zero or two** children
- **3-node** : has **two** data elements, and **zero or three** children
- **4-node** : has **three** data elements, and **zero or four** children

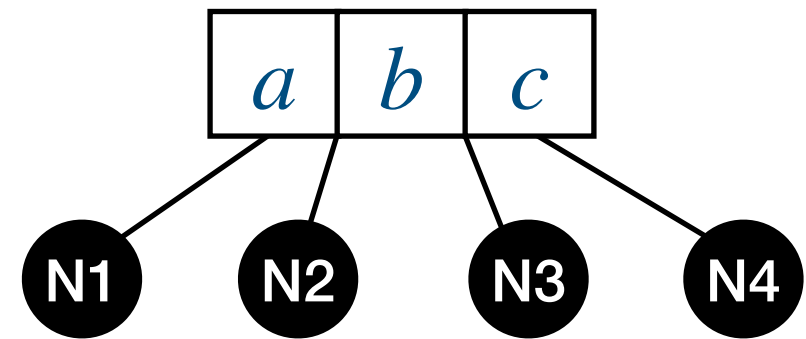
2-node



3-node



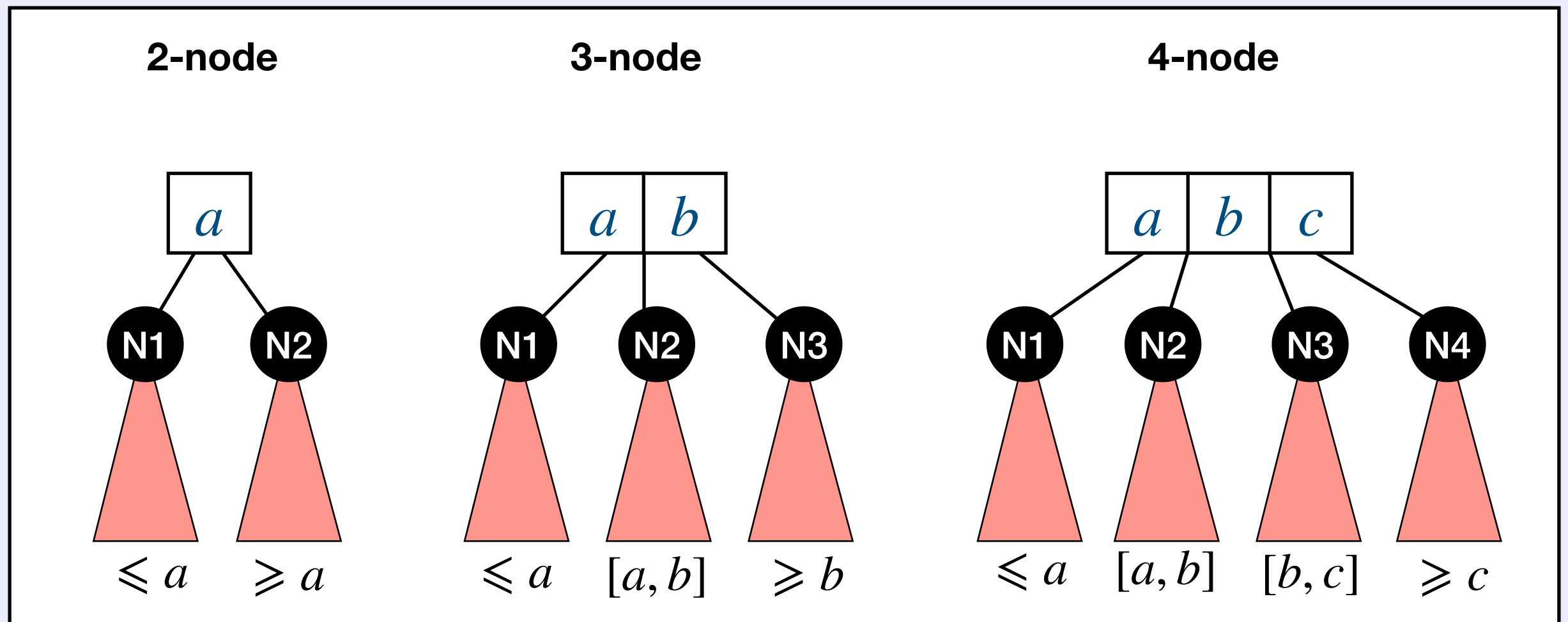
4-node



234 Trees

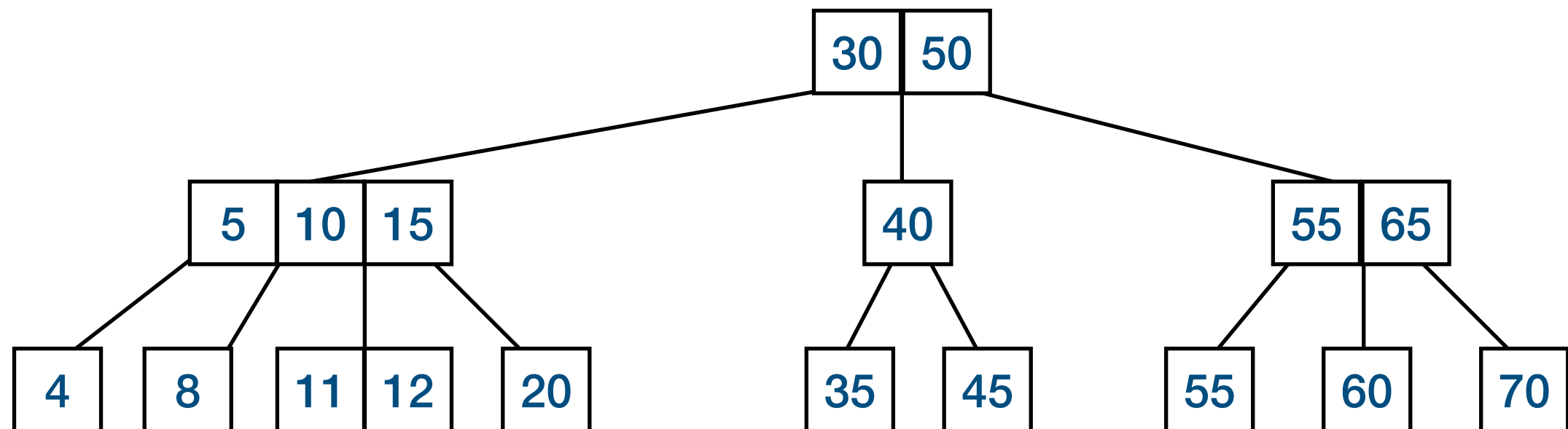
Property 1: Every node is of following type:

- **2-node** : has **one** data element, and **zero or two** children
- **3-node** : has **two** data elements, and **zero or three** children
- **4-node** : has **three** data elements, and **zero or four** children



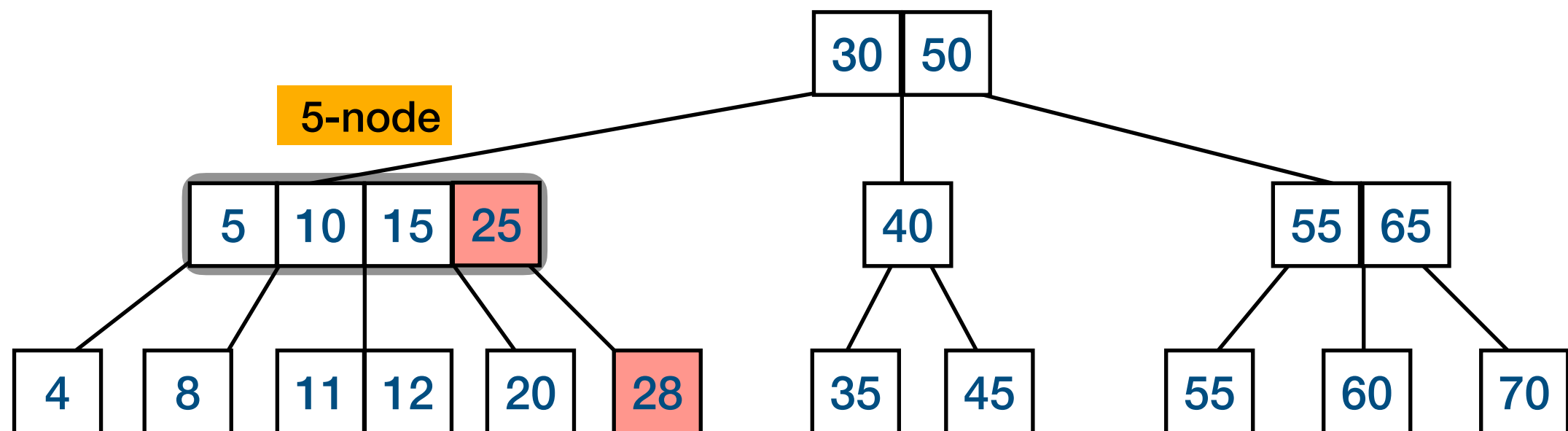
234 Trees

Property 2 : All leaves are at same level.



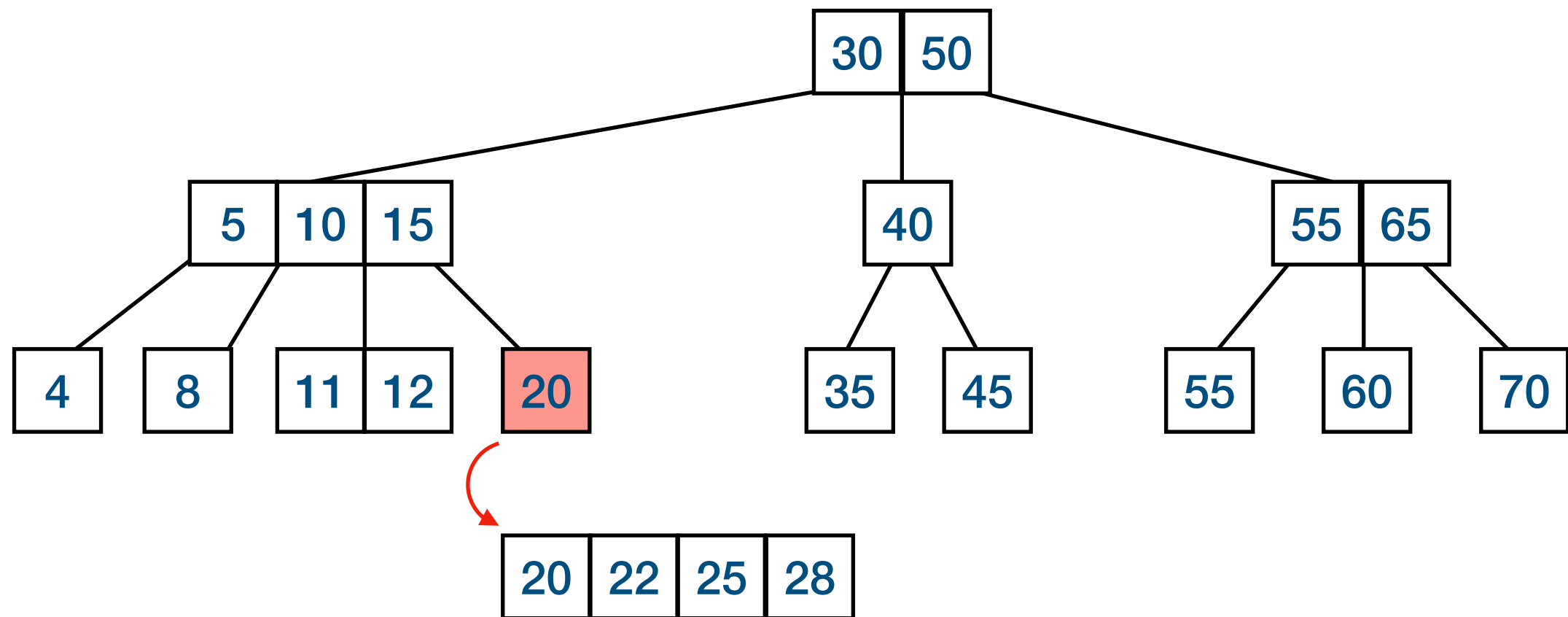
Trees with a Single Overflow

Definition: Contains exactly one 5-node.



Why study Overflow?

Reason : We can get a 5-Node after insertions



*5-Node obtained
after insertion of 22, 25, 28*

Handling Insertion

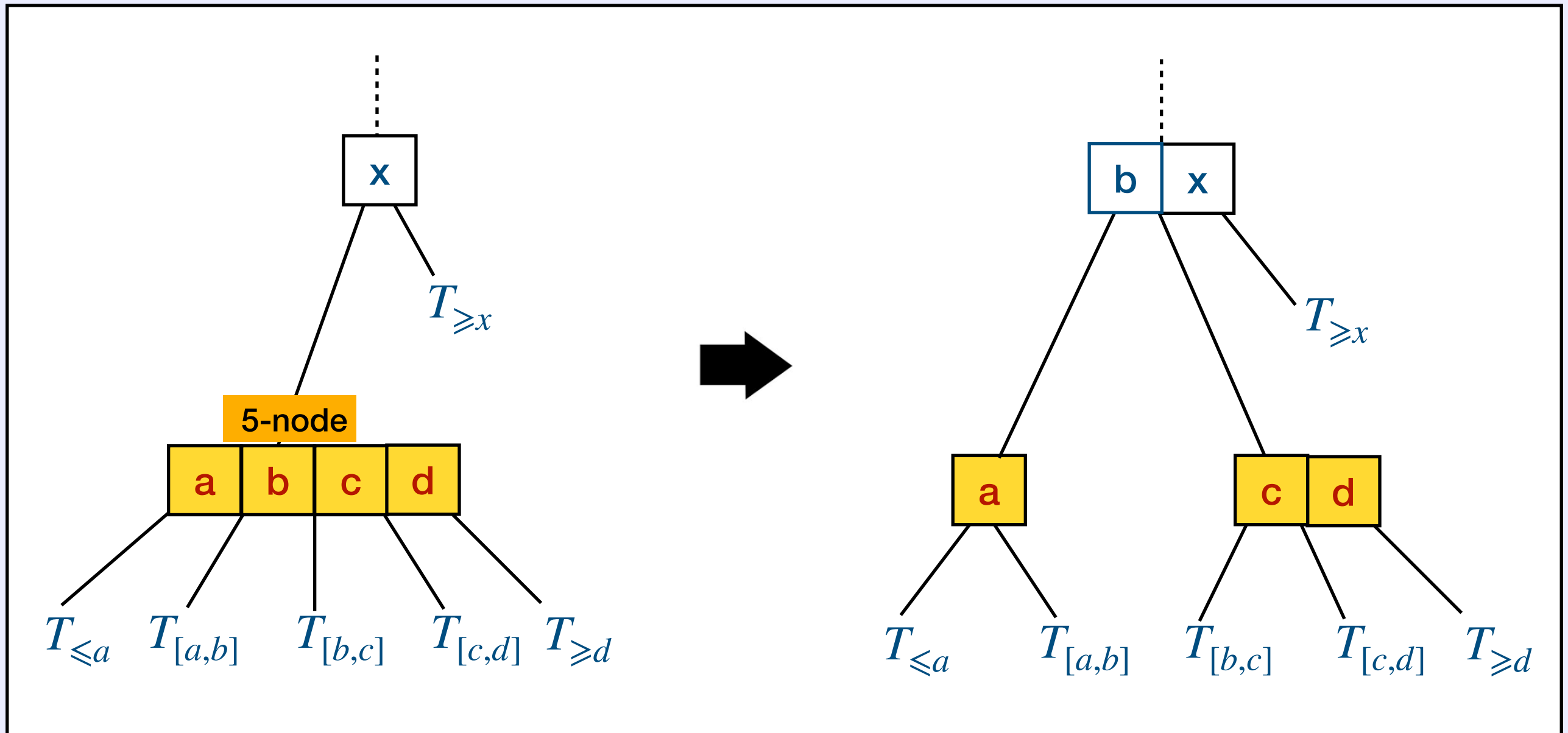
Algorithm:

Suppose a key k needs to be inserted

1. The first step is to find the leaf node, say x , that can accommodate key k without breaking the sorted ordering. This takes $O(\log n)$ time.
2. After inserting key k , the node x transitions from either 2-node to 3-node, or 3-node to 4-node, or 4-node to 5-node.
3. If x becomes a 5-node, then the insertion problem reduces to that of fixing a tree with a single overflow.

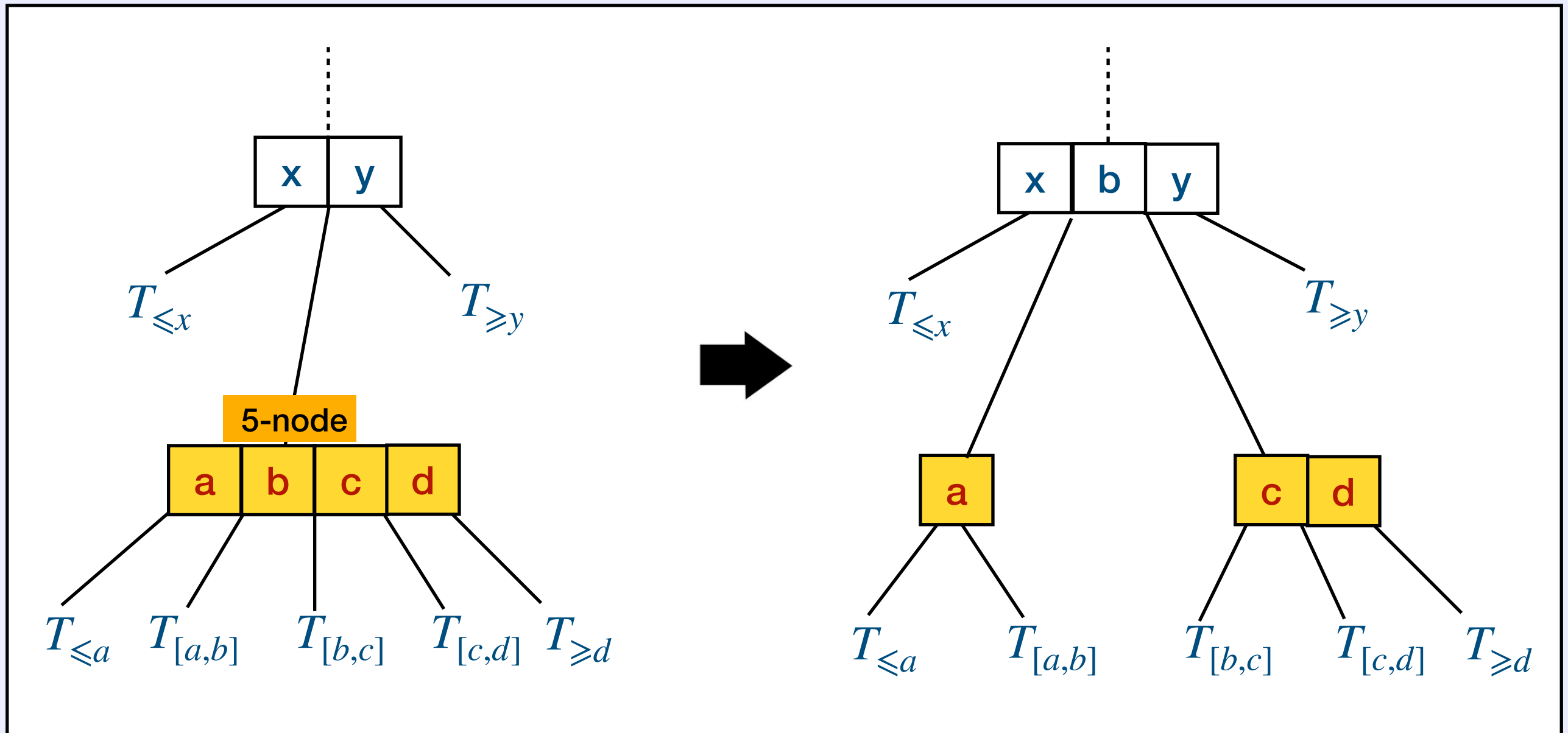
Question: How to fix trees with a Single Overflow?

Handling Overflow



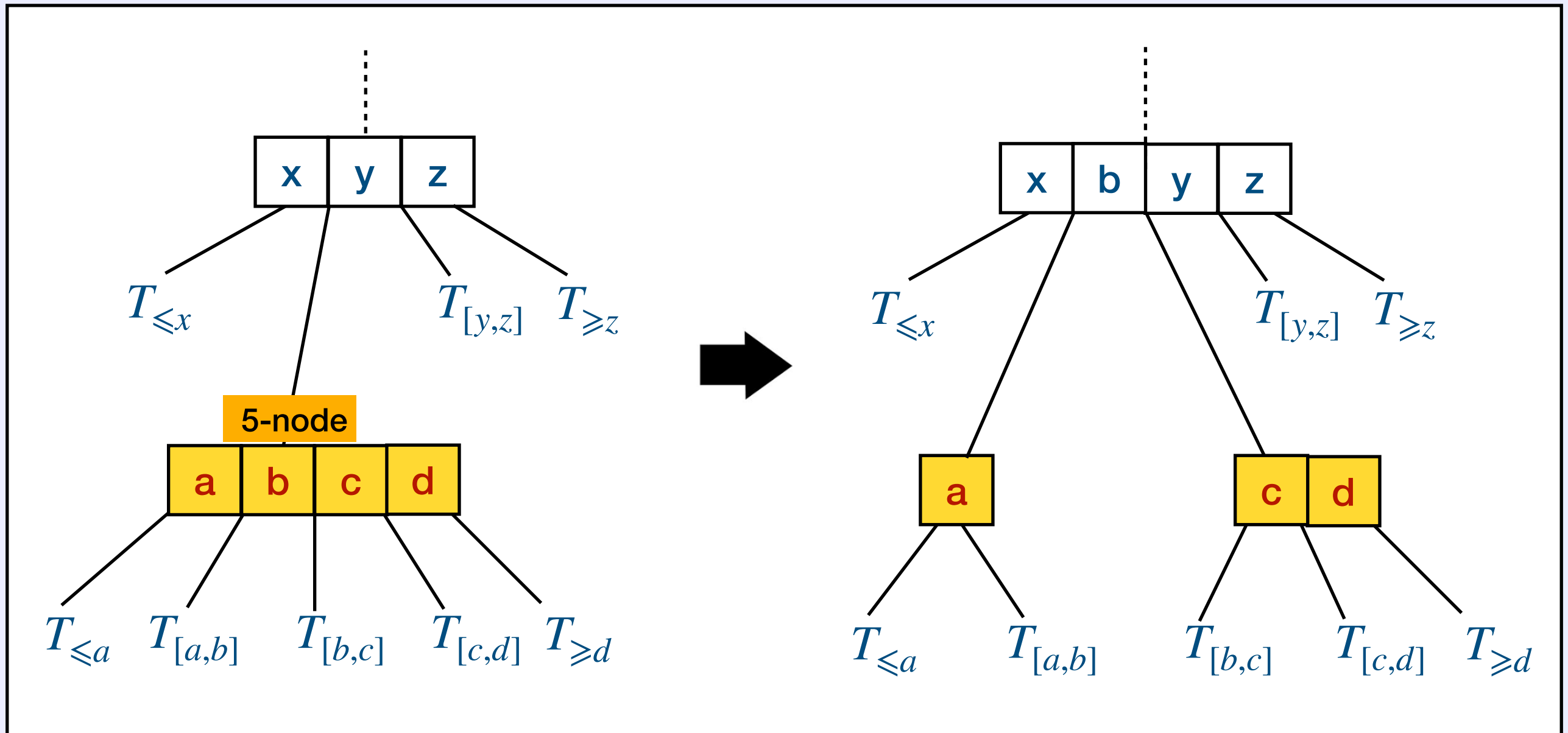
Case 1 : The parent node is originally a 2-node.

Handling Overflow



Case 2 : The parent node is originally a 3-node.

Handling Overflow



Case 3 : The parent node is originally a 4-node.

In this case the overflow is transferred to the parent node.

Handling Overflow

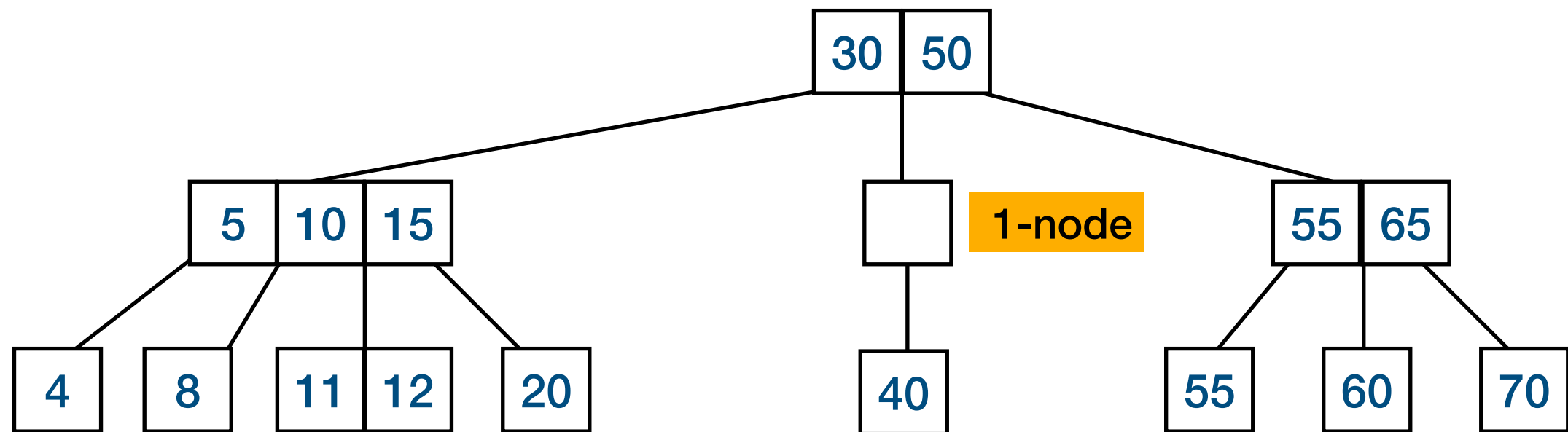
Conclusion :

Handling overflow at a single node takes $O(\log n)$ time in the worst case.

Furthermore, in the process if the overflow reaches the root then the height of the tree increases by one.

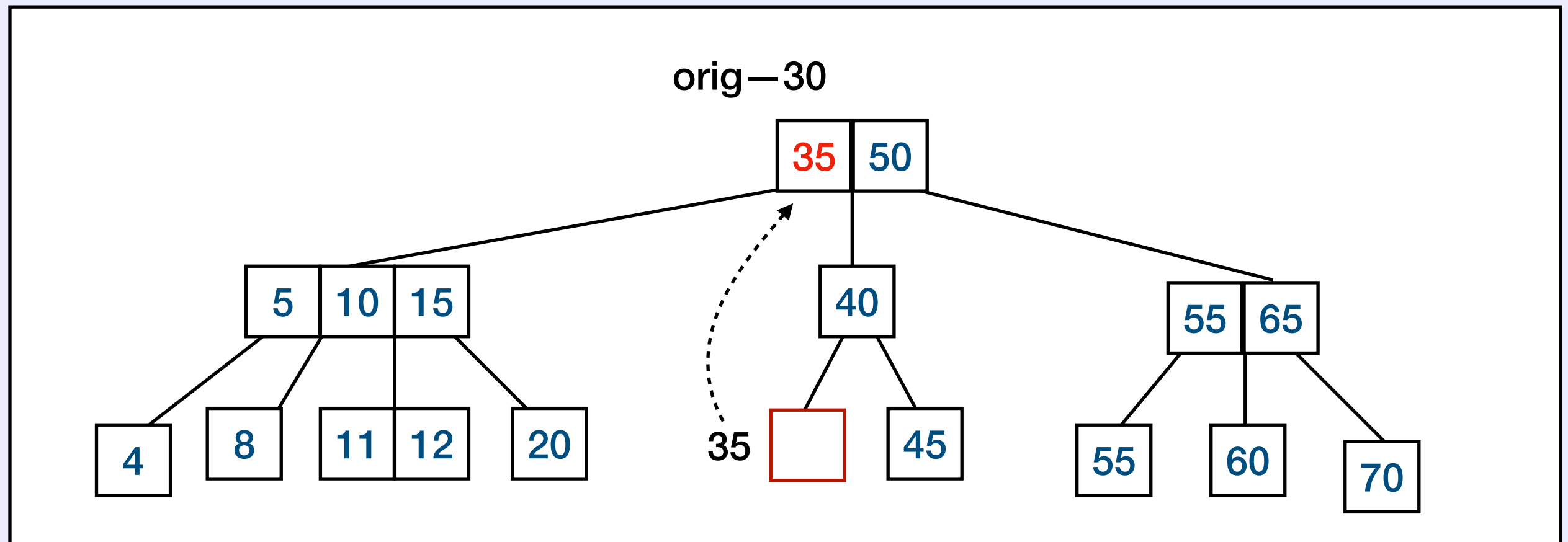
Trees with a Single Underflow

Definition: Contains exactly one “1-node”



Why study Underflow?

Reason : We can get a 1-Node after deletions

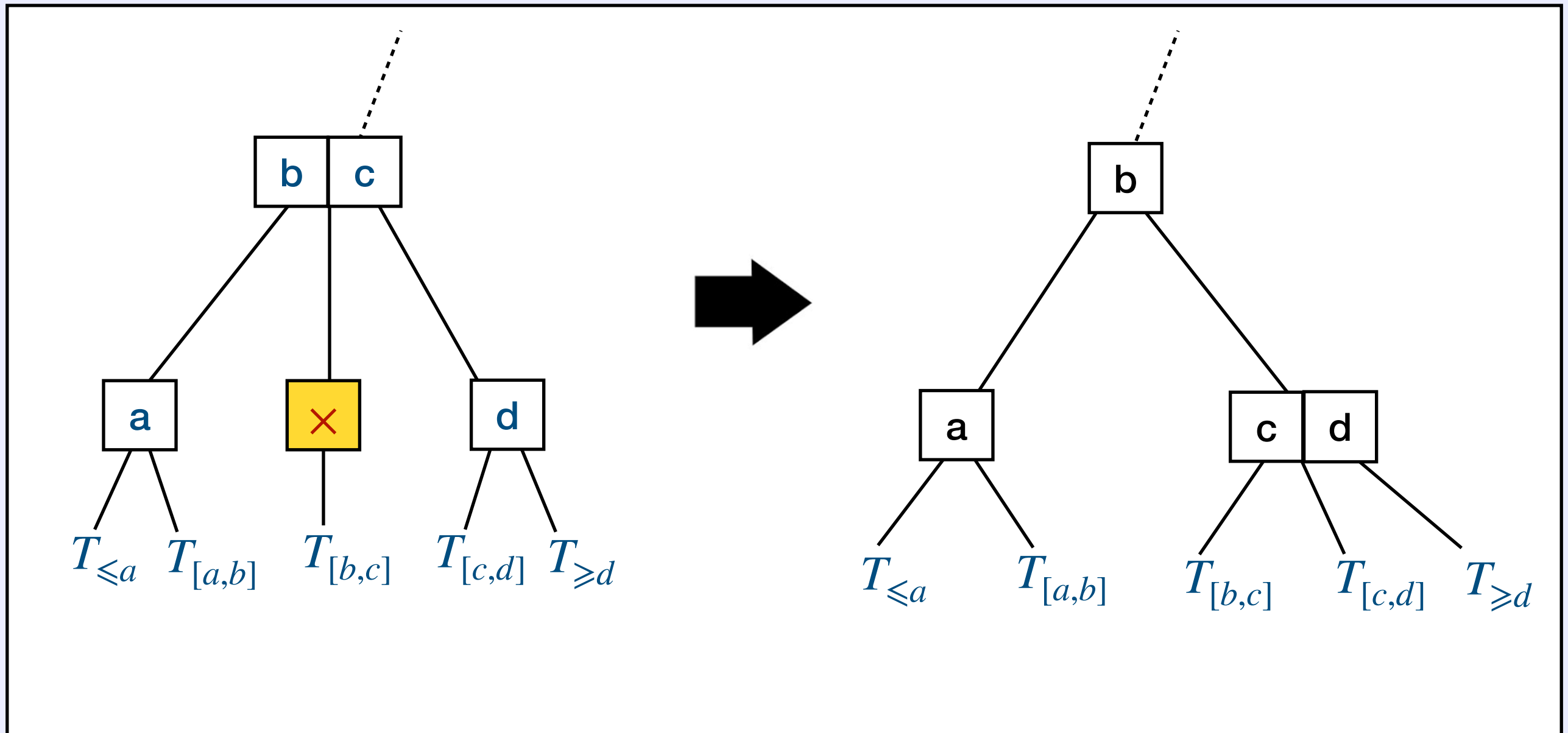


*1- Node obtained
after deletion of 30*

Algorithm:

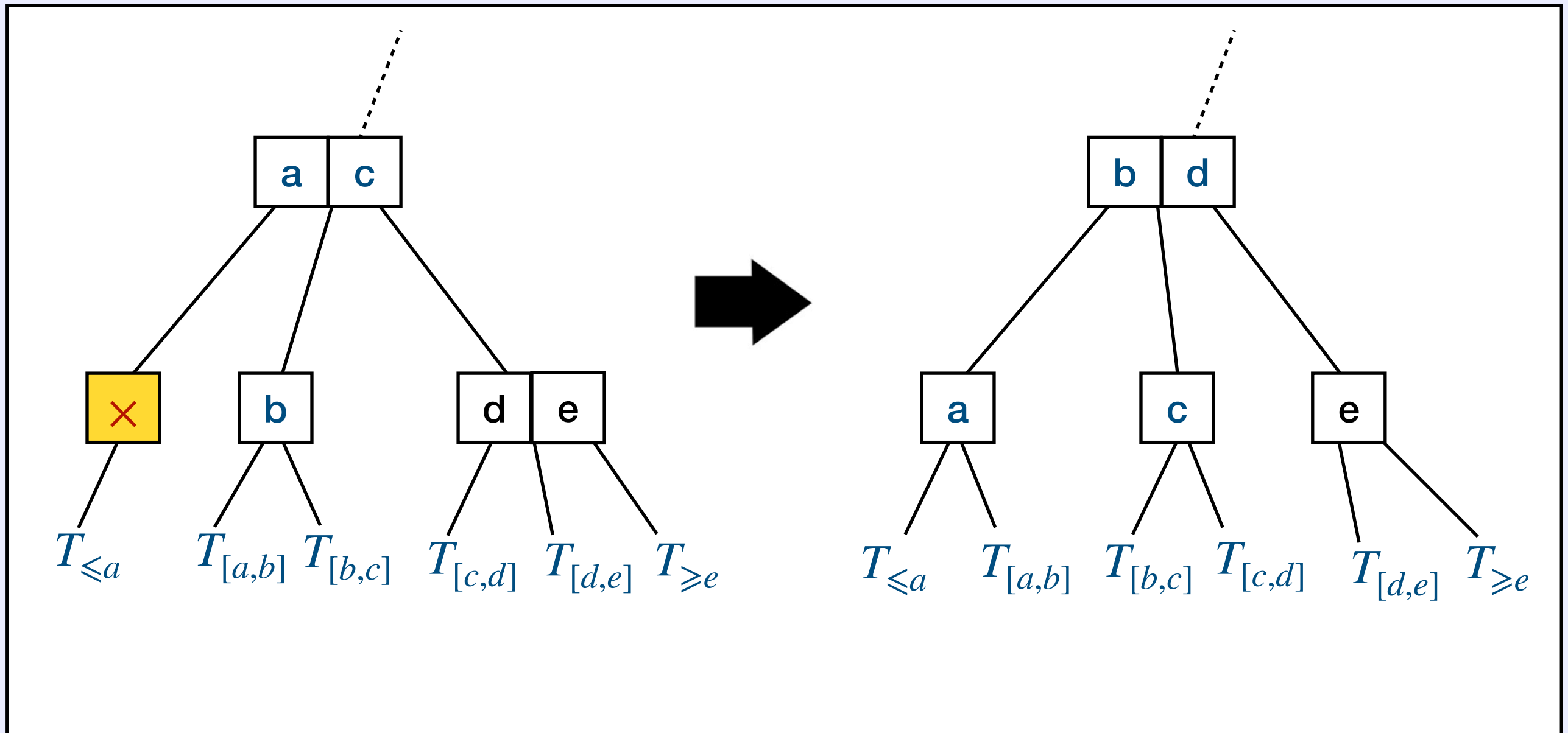
1. Suppose a key k needs to be deleted. We first search k . If k is not a leaf then we go to its successor node (say x) and swap.
2. Because successor is a leaf, we may get a 1-Node at leaf.

Handling Underflow



Case 1 : All Siblings are 2-Nodes, and the parent is either a 3-Node or 4-Node.

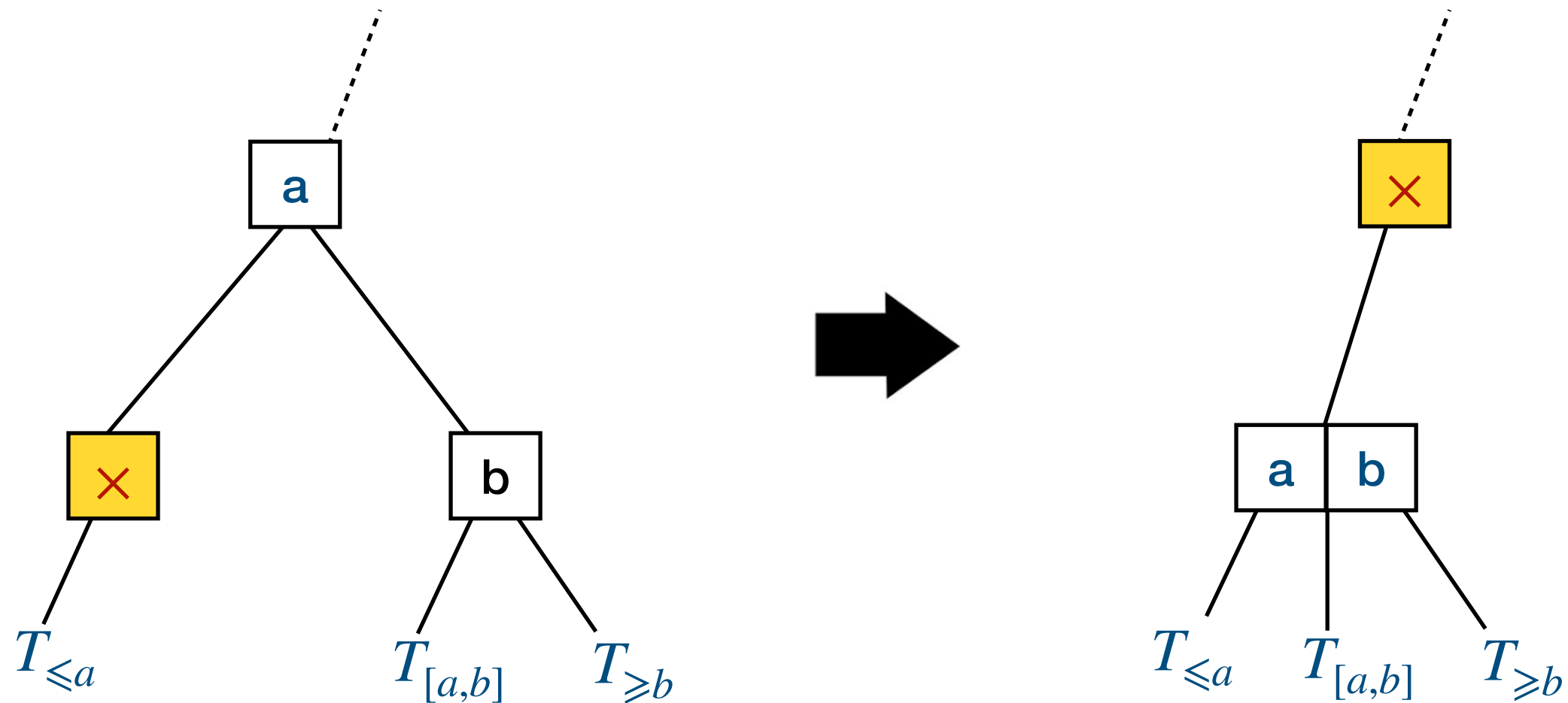
Handling Underflow



Case 2 : At least one Sibling is a 3-Node or 4-Node.

In this case structure of the parent node remains intact.

Handling Underflow



Case 3 : Parent and Sibling both are 2-Nodes.

In this case we have propagated the underflow to the parent node.

Handling Underflow

Conclusion :

Handling underflow at a single node takes $O(\log n)$ time in the worst case.

Furthermore, in the process if the underflow reaches the root then the height of the tree decreases by one.