B-link tree

Nicolás Figueroa

Macarena Oyague

Joseph Peña

Outline

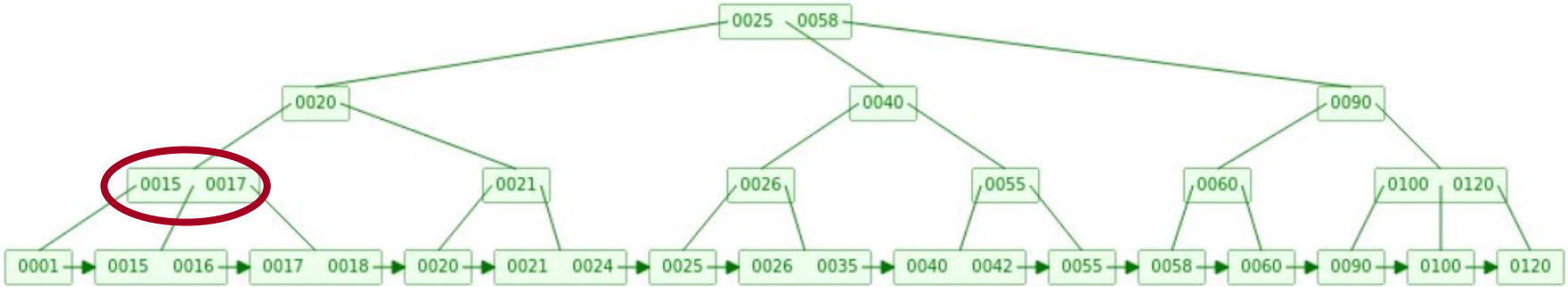
B+ tree: Race condition

Thread 1:

Search for the key 17

Thread 2:

Insert an element with key 16



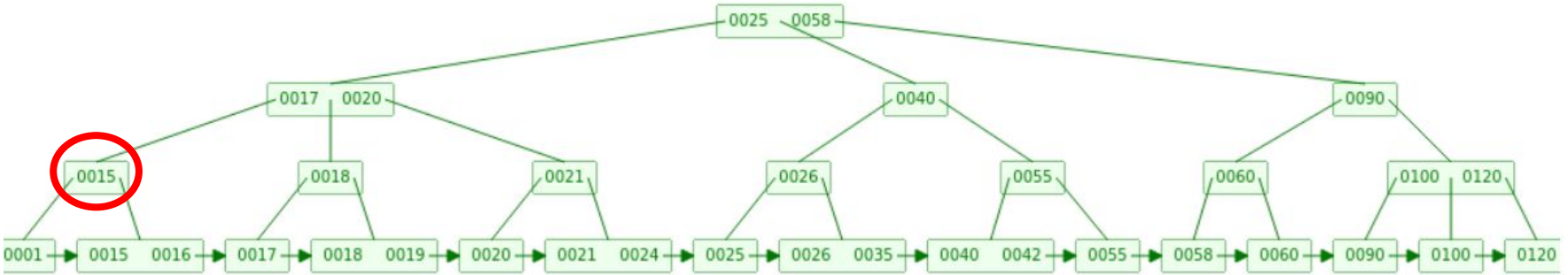
B+ tree: Race condition

Thread 1:

Search for the key 17

Thread 2:

Insert an element with key 16



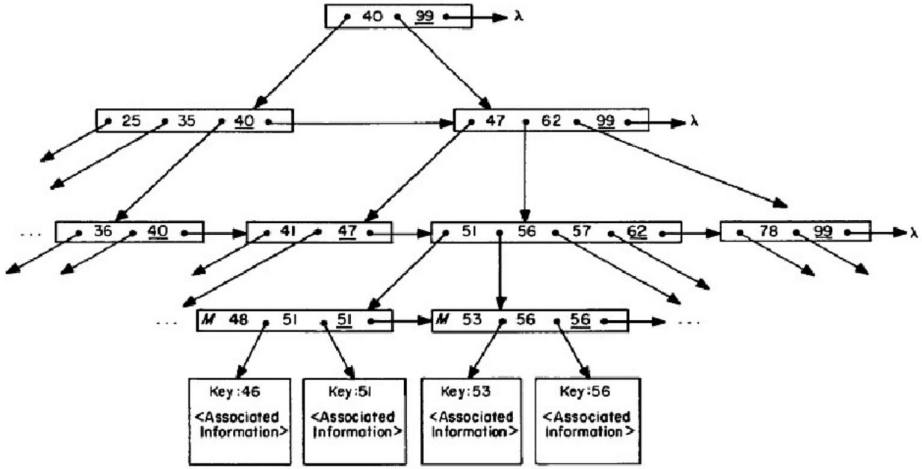
*¨**At any point,** a reader sees a **valid B(link) tree** because the tree can be implemented by considering **each node** of the tree **as a shared resource.**¨*

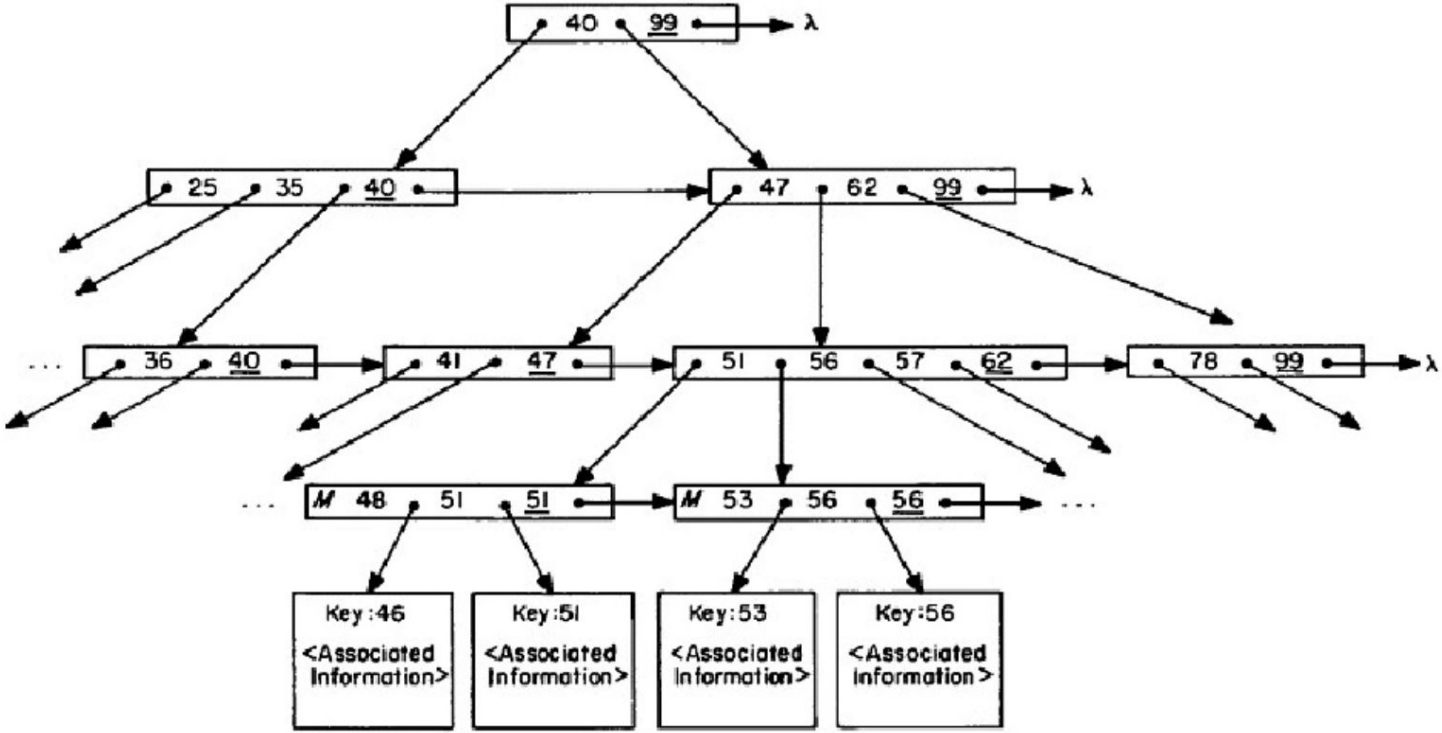
Christos Faloutsos
Professor at Carnegie Mellon University

B link tree - characteristics

B link tree

- All sibling nodes (internal nodes and leaves) are **linked together left to right**.
- An internal node with n keys has **n + 1 pointers** and the last key is known as the high key.
- Rightmost nodes: null pointers

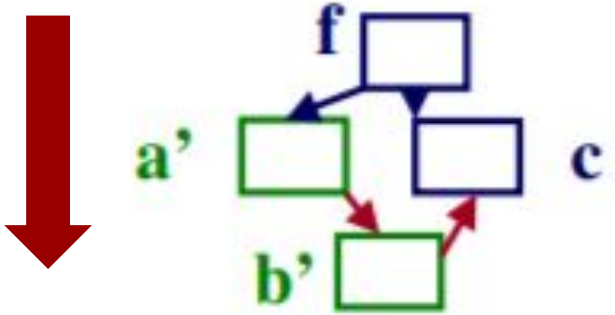



40 99 λ
25 35 40
47 62 99 λ
36 40
41 47
51 56 57 62
78 99 λ
M 48 51 51
M 53 56 56
Key:46
<Associated Information>
Key:51
<Associated Information>
Key:53
<Associated Information>
Key:56
<Associated Information>

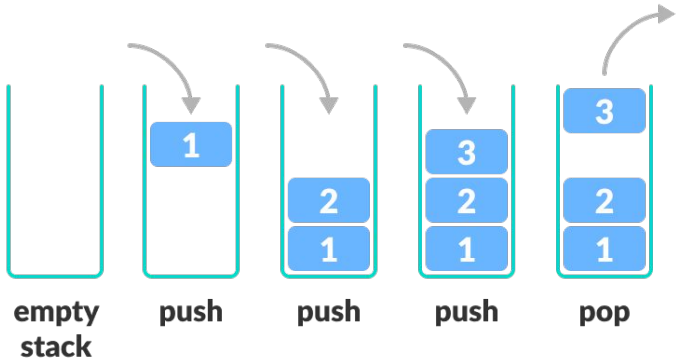
B-link tree - methods

Method: Insert

We acquire a lock on it and crab rightward until we reach the correct leaf node.

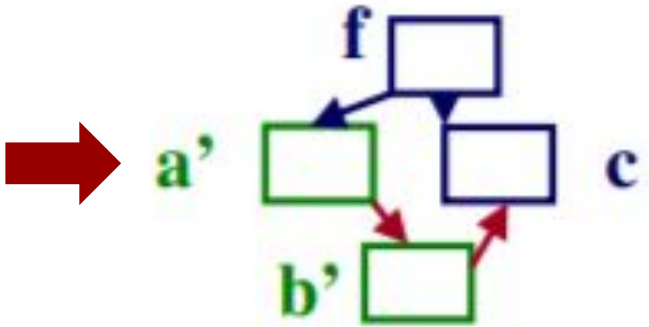


keeping track of parents in a stack.



Method: Insert

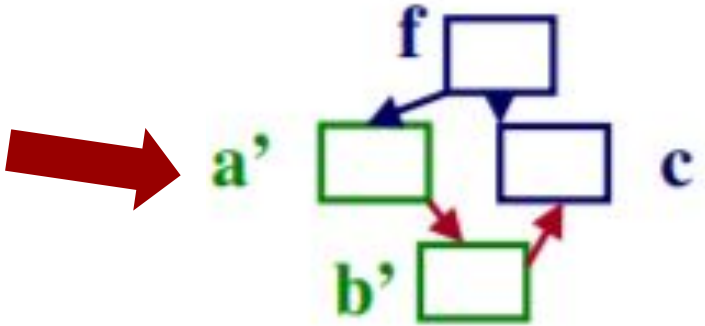
if this node is not full, then we insert and unlock the node.



if this node is full, insert into B and A.

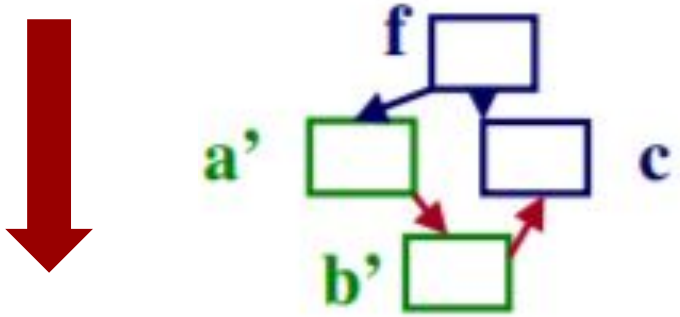
We flush B and A to disk

Next, we have to adjust the parent of A.



Method: Insert

We acquire a lock on the parent node and then crab rightward until we reach the correct parent node.
At this point, we repeat our procedure upwards through the tree.



Method: Correctness Proof

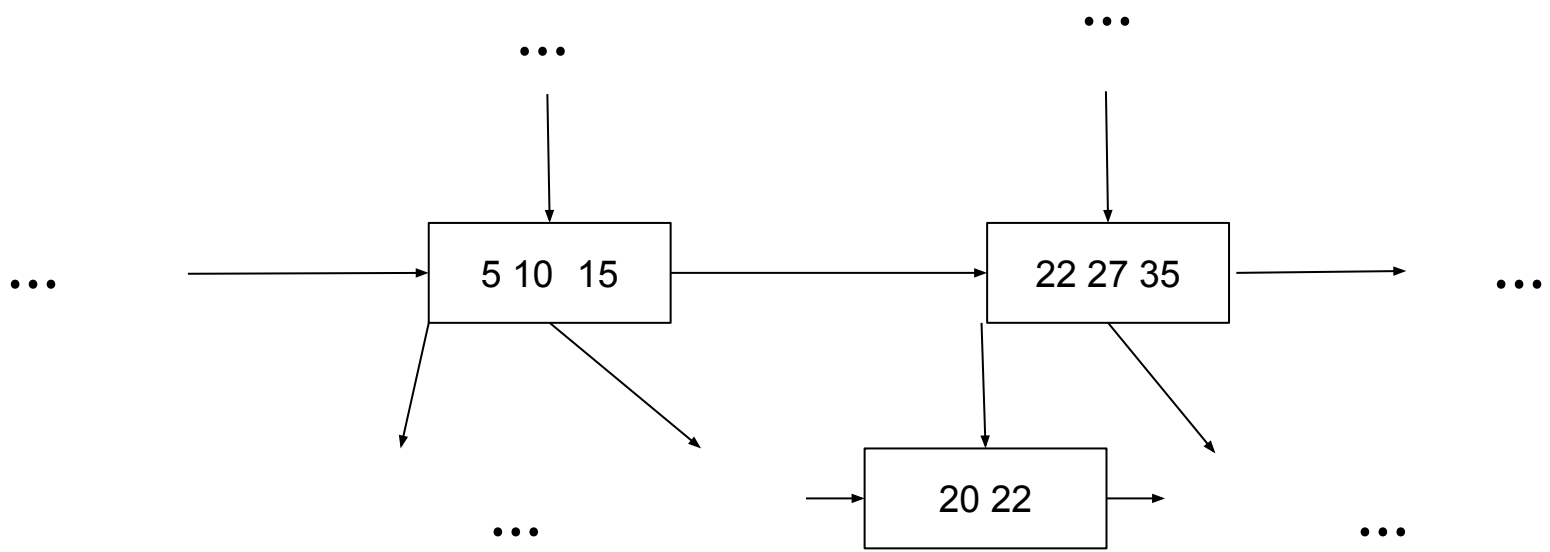
Multiple threads operating on the B-link tree **cannot deadlock**

The B-link tree **appears as a valid tree to all nodes** except the modifying thread

Method: Search

1. To search for a key xxx, we traverse the tree from root to leaf. At every internal node.
2. We compare yyy against the internal node's keys to determine which child to visit next.
3. However, we might have to walk rightward along the B-link tree to find the correct child pointer.

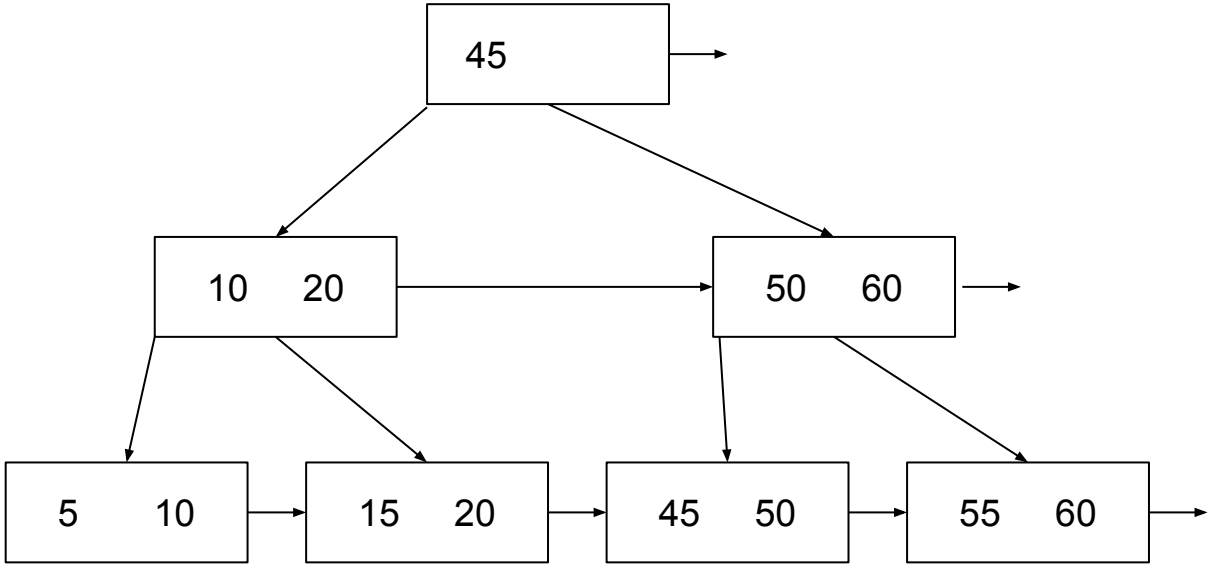
Method: Search



Method: Delete

Deletion of a key from a leaf node 20:

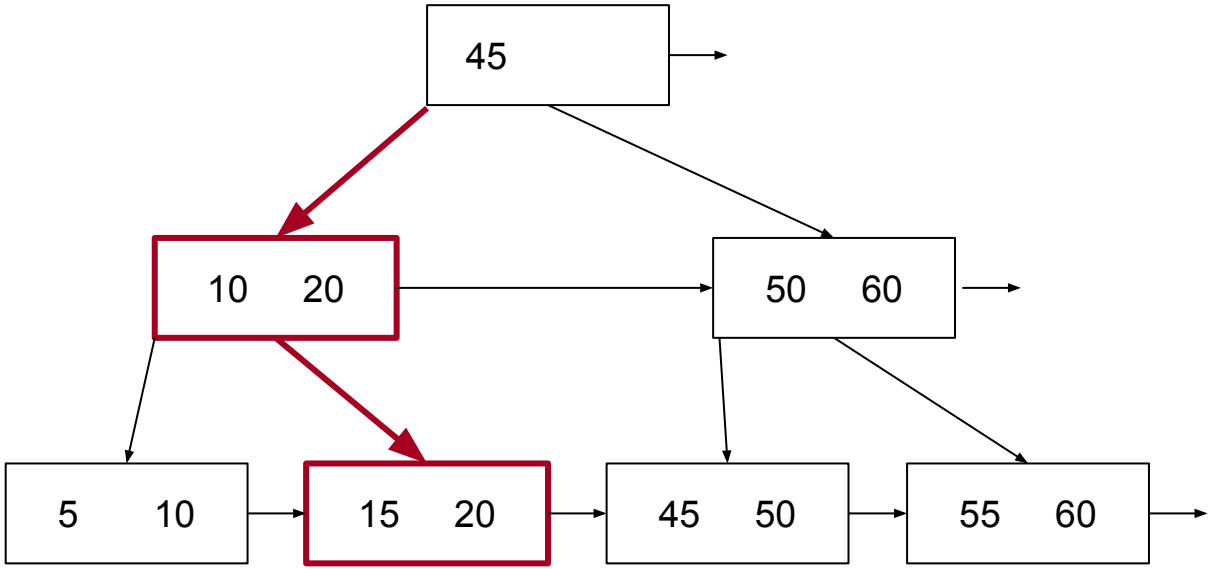
1. Search for the node
2. Lock de node
3. Read it into memory
4. Remove the value
5. Apply proof of correctness



Method: Delete

Deletion of a key from a leaf node 20:

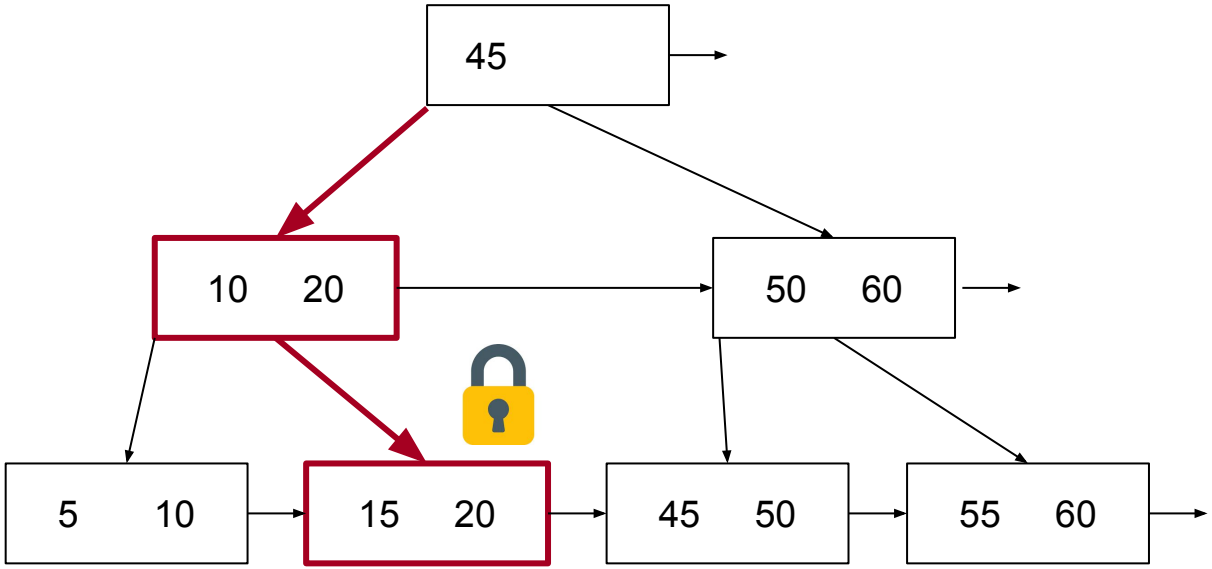
1. **Search for the node**
2. Lock de node
3. Read it into memory
4. Remove the value
5. Apply proof of correctness



Method: Delete

Deletion of a key from a leaf node:

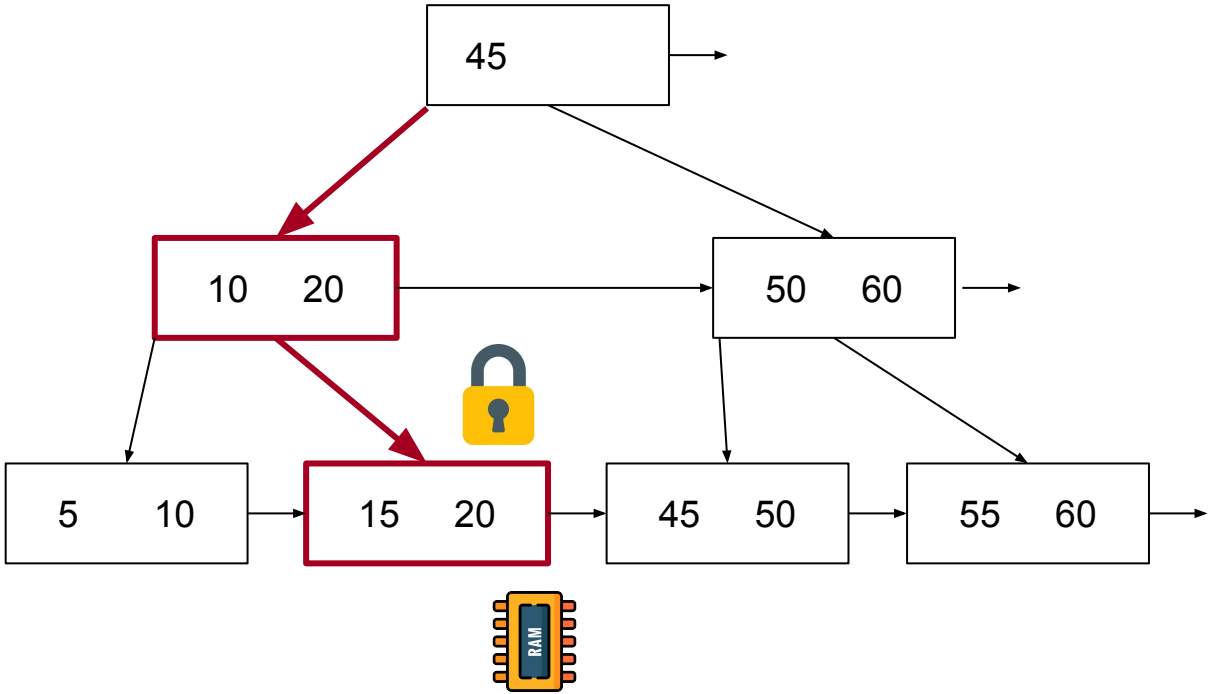
1. Search for the node
2. **Lock de node**
3. Read it into memory
4. Remove the value
5. Apply proof of correctness



Method: Delete

Deletion of a key from a leaf node:

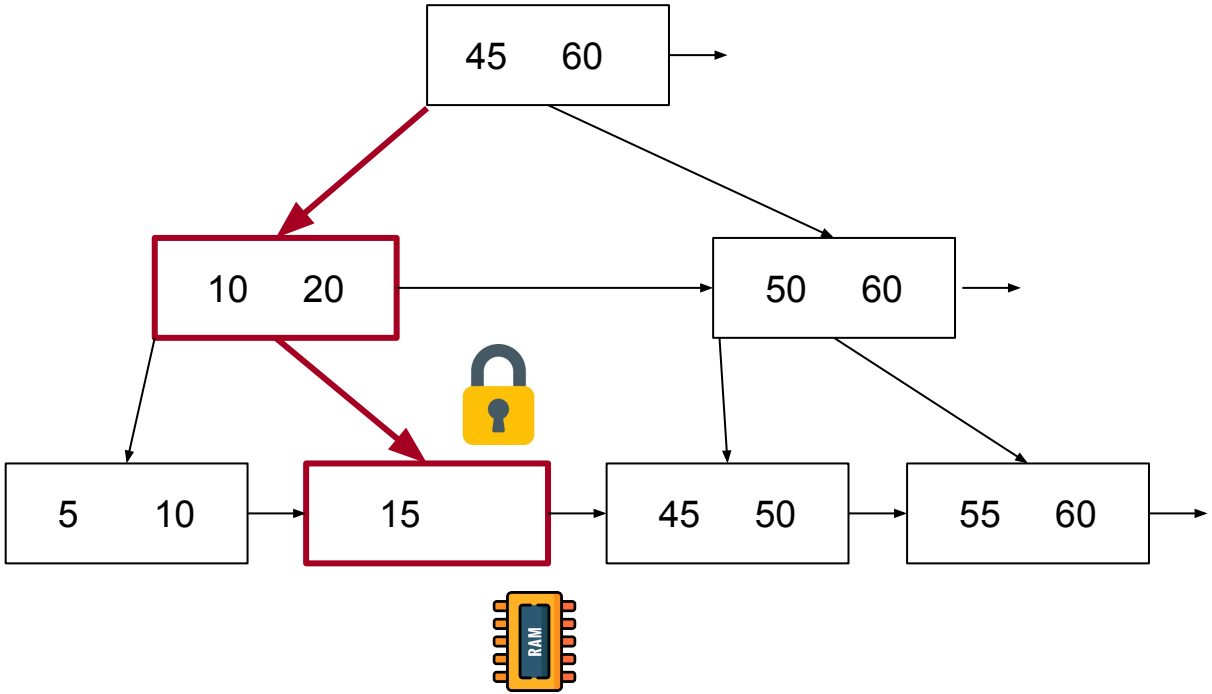
1. Search for the node
2. Lock de node
3. **Read it into memory**
4. Remove the value
5. Apply proof of correctness



Method: Delete

Deletion of a key from a leaf node:

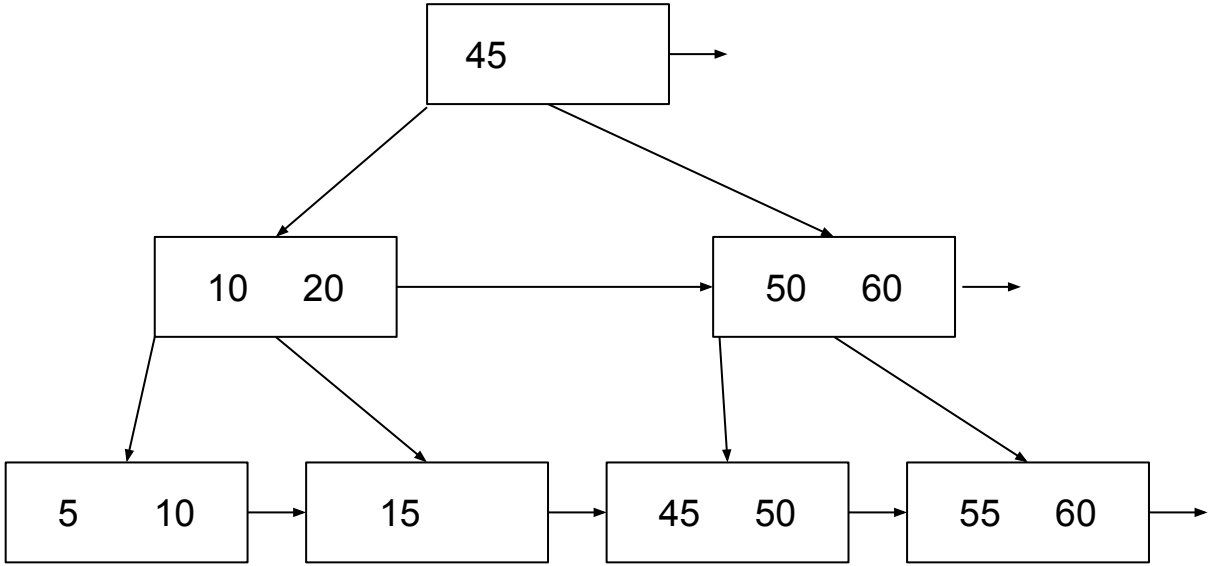
1. Search for the node
2. Lock de node
3. Read it into memory
4. **Remove the value**
5. Apply proof of correctness



Method: Delete

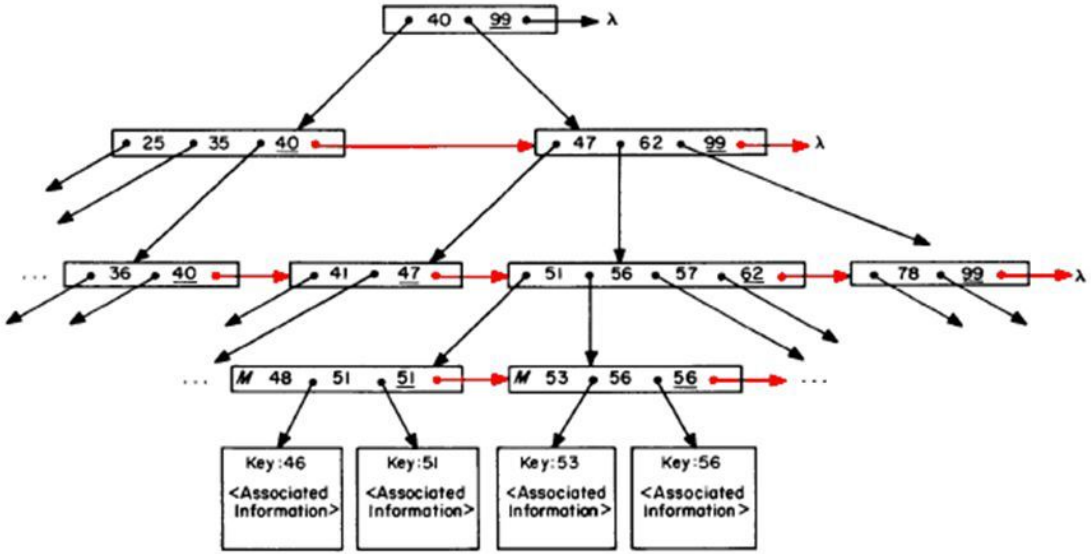
Deletion of a key from a leaf node:

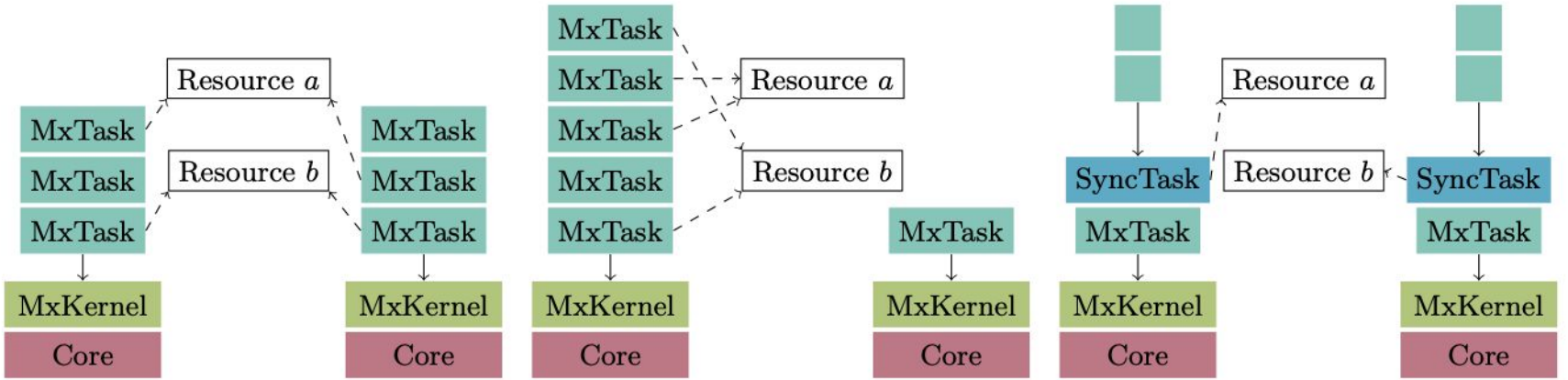
1. Search for the node
2. Lock de node
3. Read it into memory
4. Remove the value
5. **Apply proof of correctness**



Applications

- Index data in **databases** or **file systems**
- Systems with concurrent and dynamic access to data





(a) Unsynchronized access. (b) Synchronized by core mapping. (c) Synchronized by Synchronize-Task.

Resource: http://ceur-ws.org/Vol-2126/paper9.pdf

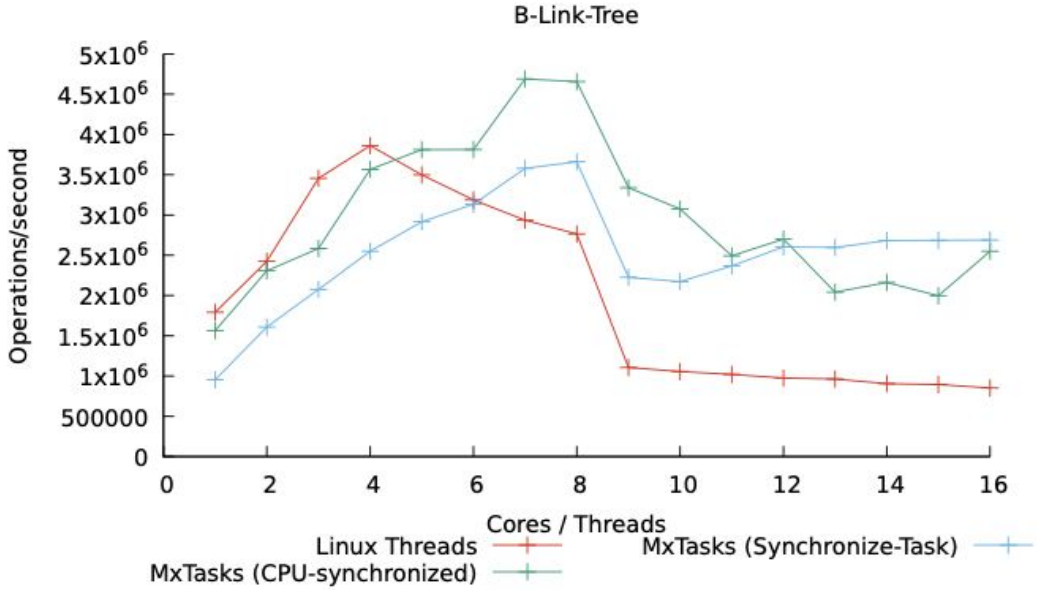


Figure 3: Results of the B-link-tree.

Resource: http://ceur-ws.org/Vol-2126/paper9.pdf

Time complexity

Search	**O(logn)**
Insert	**O(logn)**
Delete	**O(logn)**

*** Worst, average and best case**

Advantages

- **The performance with concurrent operations is better than others implementations of b+ tree.**

- **That synchronization like locking has to be avoided both for Operating Systems (OSs) and applications running on top, including Database Management Systems (DBMSs).**

Disadvantages

- **The use of concurrent programming is a delicate application due to the different problems that can be caused by a wrong implementation. For example:**
- **Livelock:**
  - **This can happen if a process never terminates because it keeps having to follow link pointers created by other processes.**
  - **This might happen in the case of a process being run on a (relatively) very slow processor in a multiprocessor system.**

Conclusions

Conclusions

	B-Tree	B+-Tree	B-Link-Tree
Stores values in leaf nodes only, inner nodes point the way down to child nodes using keys.	NO	YES	YES
Every node contains a high key, which indicates the highest key that node will hold, and a link to the right sibling that allows sequential processing of the inner and leaf nodes.	NO	NO	YES
Use only three nodes at most to make operations on it.	NO	NO	YES