University of Dublin
Trinity College

B+-Trees

Owen.Conlan@cs.tcd.ie

B+-Trees

most common variant of B-tree index is B+- tree

data pointers are stored *only at the leaf nodes*

structure of leaf nodes differs from internal nodes

leaf nodes have entry for *every* value of the search field along with a pointer to the record (or the block that contains the record)

for non-key search field, pointer points to a block containing pointers to the data file records, creating extra level of indirection

leaf nodes generally linked together to provide ordered access on the search field to the records

B+-Trees

Structure of internal nodes of a B+ - tree:

each internal nodes is of the form:

$< A_1, K_1, A_2, K_2, ..., A_{k-1}, K_{k-1}, A_k >$

where $k \leq m$ and each A_i is a tree pointer

within each internal node, $K_1 < K_2 < .... K_{k-1}$

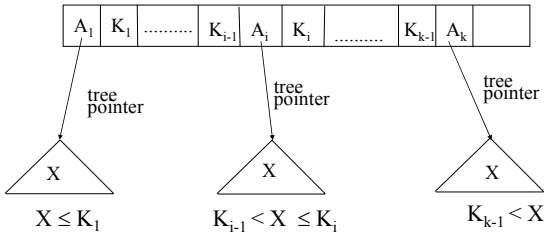
for all search field values X in the sub-tree pointed at by A_i, we have

$K_{i-1} < X < K_i$ for $1 < i < k$,

$X \leq K_i$ for i=1, and

$K_{i-1} < X$ for $i = k$

Structure of Internal nodes of B+-Trees



Internal node of a B+ - tree

Structure of Internal nodes of B+-Trees

each internal node has at most m tree pointers

each internal node, except the root, has at least CEIL [m/2] tree pointers; the root node has at least 2 tree pointers if it is an internal node

an internal node with k pointers, $k \leq m$, has k-1 search field (key) values.

Structure of Leaf nodes of B+-Trees

Each leaf node is of the form

$< <K_1, R_1>, <K_2, R_2>, ... <K_{k-1}, R_{k-1}>, R_{next} >$

where $k \leq m$, each R_i is a data pointer, and R_{next} points to the next *leaf node* of the B+ - tree

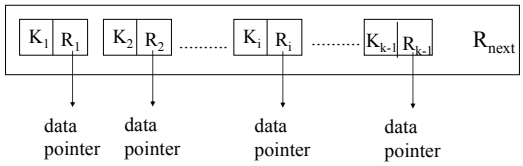
within each leaf node, $K_1 < K_2 < ..... K_{k-1}$, $k \leq m$

Each R_i is a *data pointer* that points to the records whose search field value is K_i or to a file block containing the record (or a block of record pointers that point to records whose search field value is K_i if the search field is not a key)

Each leaf node has at least CEIL[m/2] -1 values

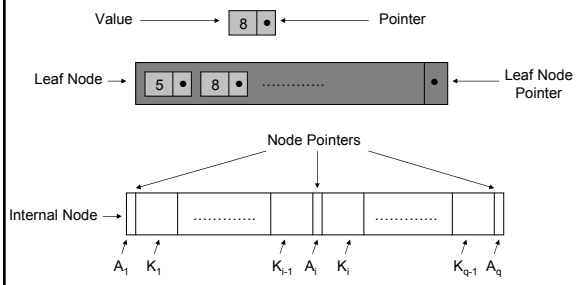
All leaf nodes are at the same level

Structure of Leaf nodes of B+-Trees



Leaf node of a B+ - tree

B+-Trees Legend



Insertion into a B+-Tree

Insert 8

Insertion into a B+-Tree

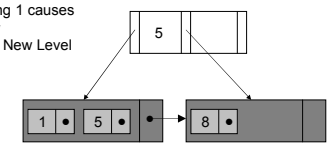
Insert 5

Example 1 - Insertion into a B+-Tree

Insert 1

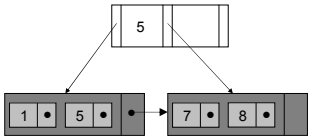
Insertion into a B+-Tree

- Inserting 1 causes overflow
- Create New Level



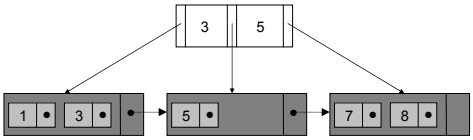
Insert 7

Insertion into a B⁺-Tree



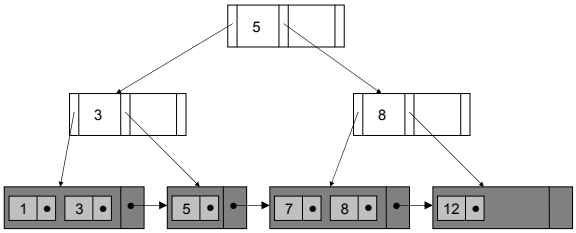
Insert 3

Insertion into a B⁺-Tree



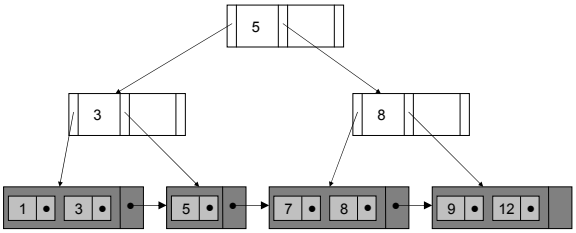
Insert 12

Insertion into a B⁺-Tree



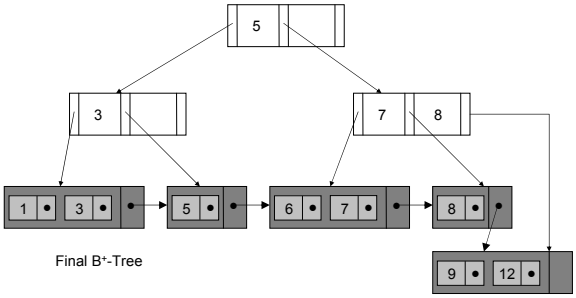
Insert 9

Insertion into a B⁺-Tree



Insert 6

Insertion into a B⁺-Tree



Final B⁺-Tree

B⁺-Tree Variations

- Many variations
  - B-trees
  - B⁺-trees
  - B*-trees
- Common modifications
  - Change the fillfactor from 0.5 to 1.0
  - Allow a node to become empty before merging

Review

B$^+$-Tree is the most common variant of the B-Tree

It has two types of nodes

- Internal nodes
- Leaf nodes

Is more efficient than a B-Tree (see handout)