Module 4: Index Structures

Lecture 13: Index structure

Prev topic | Next topic | Prev page | Next page

The Lecture Contains:

- Index structure
- Binary search tree (BST)
- B-tree
- B+-tree
  - Order

Index structure

- Five important questions when deciding on an index structure:
    - Type of data
        - Points or shapes or intervals
        - Discrete or continuous
        - Dimensionality and/or distance
    - Queries on data
    - Index data or index space where data resides
    - Static or dynamic
    - Memory (small) or disk-resident (large)

Index structure

- Five important questions when deciding on an index structure:
    - Type of data
        - Points or shapes or intervals
        - Discrete or continuous
        - Dimensionality and/or distance
    - Queries on data
    - Index data or index space where data resides
    - Static or dynamic
    - Memory (small) or disk-resident (large)
- Properties:
    - Fast access
    - Update overhead
    - Small size
    - Correctness

Binary search tree (BST)

- Binary tree
- Unbalanced
- If a node has key k, then every key l in left subtree and every key r in right subtree follows the properties:
  - $l \leq k$
  - $r > k$

Binary search tree (BST)

- Binary tree
- Unbalanced
- If a node has key k, then every key l in left subtree and every key r in right subtree follows the properties:
  - $l \leq k$
  - $r > k$
- Searching, insertion, deletion takes $O(\lg N)$ time on average.
- Worst-case times are $O(N)$ due to unbalanced property.
- AVL-trees, Red-black trees, etc. are balanced.

B-tree

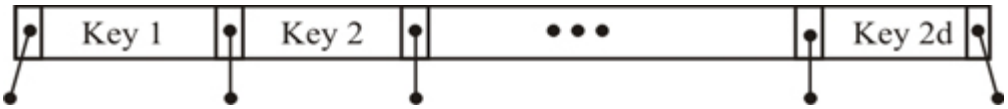
- A B-tree of order m has the following properties:
    - Leaf nodes are in same level (balanced) and contain no data
    - Internal nodes (not the root) have between m and 2 m keys
    - Root has at least 1 key
    - An internal node with r keys have $r + 1$ children
    - Child pointers in leaf nodes are null
- Branching factor = $m + 1$

B+-tree

- Balanced
- All data are in the leaf nodes
- Leaf nodes have $m \leq r \leq 2m$ keys and $r+1$ data pointers
- Internal nodes contain $m \leq r \leq 2m$ keys and $r+1$ child pointers
- Keys dene range of values for children
- Often siblings are connected by pointers to avoid parent traversal

Order

- Branching factor or order determined by page size, size of key and size of pointer



- Page size = 4 kB
- Size of key = 8 bytes
- Size of pointer = 4 bytes
- If order is m, then $8 \times 2m + 4 \times (2m + 1) = 4 \times 1024$
- Therefore, $m = 170$
- A tree of height 3 can, therefore, store 5×10^6 records and that of height 4 can store 8×10^8 records.

Module 4: Index Structures

Lecture 13:

B+-tree

- Balanced
- All data are in the leaf nodes
- Leaf nodes have $m \leq r \leq 2m$ keys and $r+1$ data pointers
- Internal nodes contain $m \leq r \leq 2m$ keys and $r+1$ child pointers
- Keys dene range of values for children
- Often siblings are connected by pointers to avoid parent traversal

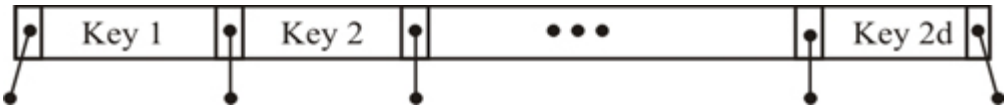
Previous Next

Module 4: Index Structures

Lecture 13:

Order

- Branching factor or order determined by page size, size of key and size of pointer



- Page size = 4 kB
- Size of key = 8 bytes
- Size of pointer = 4 bytes
- If order is m, then $8 \times 2m + 4 \times (2m + 1) = 4 \times 1024$
- Therefore, $m = 170$
- A tree of height 3 can, therefore, store 5×10^6 records and that of height 4 can store 8×10^8 records.

Previous Next