B-Tree

B-Tree is a multi-way search tree commonly used in database systems where data is stored in external disks.

Consider a disk where all data are being stored. Disk reads and writes its data one block at a time, which are usually 512, 1K or 2K Bytes. Since, reading a block takes a significant amount of time (10^6 times that of in-memory read), a shallow tree structure is implemented which minimizes number of disk reads at the expense of many in-memory operations.

Order of a B-Tree B, represents the maximum number of children for each nodes. A B-Tree of order 3 is a 2-3 tree. Each node in a B-Tree is a block. Unlike Binary Search Tree which grows from leaves, B-Tree grows from the root. A B+ Tree is a variant of B-Tree in which all data records are stored only in leaves. This version of B-Tree is implemented on the module $b\_tree.py$

B-Tree has following properties that it must satisfy:

- Non-leaf nodes (except root node) have $\lceil B/2 \rceil \leq$ Number of children $\leq B$. Nodes must be at least half full.
- Root node is either a leaf or has $2 \leq$ # of children $\leq B$.
- Non-leaf nodes store keys, one less than the # of children. Keys are sorted in the node. They guide the search toward leaves.
- All leaves are on the same depth (is always balanced). Leaves must also be at least half full.

(Rule second and fourth must be relaxed for initial insertions)

Calculate the Order of a B-Tree

Suppose that:

- A block has size S byte.
- The keys being stored has size K byte.
- Each data item in the disk have size D byte, including keys.
- The address of a block on the disk is A byte.

Number of data items per leaf = Q = S/D
Maximum number of children = Order of B-tree = B = $(S - A)/(A + K) + 1$

For 4 Terabyte disk:

- Block S = 1K byte (let's say)
- Key K = 32-bit = 4 bytes

- Data D = 12 bytes (let's say) + 4 bytes = 16 bytes (including key)
- Address A = 32-bit = 4 bytes. (Number of blocks = 4TB/1K = 4G blocks)

$$Q = 1024/16 = 64$$
$$B = (1024 - 4)/(4 + 4) + 1 = 128$$

Thus, the order of the B-Tree is 128 and each leaf can save up to 64 data items.

Generating bounds on height of B+ Tree

Consider a B+ Tree (records stored only in leaves) of order B and height h containing N nodes.

Minimum number of records needed for it to be a B+ Tree:

$$N >= (\frac{Q}{2}) \times \lceil (\frac{B}{2}) \rceil^{h-1} \times 2$$

This happens when,

- Root has 2 children
- Each of the two sub-tree of height $h - 1$ has $\lceil (B/2) \rceil^{h-1}$ nodes at height $h - 1$. Each internal nodes has only $\lceil (B/2) \rceil$ children.
- Each $\lceil (B/2) \rceil^{h-1} \times 2$ leaf nodes has $Q/2$ records

Maximum number of records it can hold:

$$N <= Q \times B^h$$

This happens when,

- Each non-leaf nodes have B children including root
- Each leaf nodes has Q records

Upper bound on height is given as,

$$log(\frac{N}{Q}) >= (h - 1) log(\lceil \frac{B}{2} \rceil)$$

$$h <= \frac{log(\frac{N}{Q})}{log(\lceil \frac{B}{2} \rceil)} + 1$$

Lower bound on height is given as,

$$log(\frac{N}{Q}) <= h \times log(B)$$

$$log(\frac{N}{Q}) <= h \times log(\frac{B}{2} \times 2) = h \times (log(\frac{B}{2}) + 1)$$

$$h >= \frac{log(\frac{N}{Q})}{log(\frac{B}{2}) + 1}$$

Thus,

$$\frac{log(\frac{N}{Q})}{log(\frac{B}{2}) + 1} =< h <= \frac{log(\frac{N}{Q})}{log(\lceil \frac{B}{2} \rceil)} + 1$$

For the special case when $Q = B$ and B is even,

$$h <= log_{\lceil B/2 \rceil} N + 1 - log_{\lceil B/2 \rceil} B = log_{\lceil B/2 \rceil} N + 1 - (1 - log_{\lceil B/2 \rceil} 2)$$

$$h <= log_{\lceil B/2 \rceil} \frac{N}{2}$$

Height of a B-Tree is $O(log_{\lceil B/2 \rceil} N)$

Consider a B-Tree (records stored in all nodes) of order B of height h containing N nodes. Minimum number of records needed for it to be a B-Tree:

$$N >= 1 + (\lceil B/2 \rceil - 1) \times \sum_{i=1}^{h} 2 \times \lceil B/2 \rceil^{i-1}$$

This happens when,

- Root node stores 1 record
- Each node stores $(\lceil B/2 \rceil - 1)$ records

Upper bound of height is given by,

$$N >= 1 + 2(\lceil B/2 \rceil - 1) \times (\frac{\lceil B/2 \rceil^h - 1}{\lceil B/2 \rceil - 1})$$

$$N >= 2\lceil B/2 \rceil^h - 1$$

$$h <= log_{\lceil B/2 \rceil}(\frac{N+1}{2})$$

Height of a B-Tree = $O(log_{\lceil B/2 \rceil} N)$