CS3563 : DBMS II
Assignment 3

Prof. Dr. Manohar Kaul

April 20, 2020

Storage Management

Problem 1.

Consider a hard disk with 4 platters with two recording surfaces. There are 8192 cylinders in the disk and each track in a cylinder has 512 sectors, each with capacity 512B. Each sector has the following addressing scheme <recording surface number, track number, sector number> and indexing starts at 0. The last position of the disk head is <3, 4095, 127>. The spindle of the disk rotates at 7200 RPM and the average seek time is 5ms. Mention all the assumptions made with proper justification.

(a) What is the size of the disk? [4]

(b) A file of size 1000 MB has to be stored. How will this be ideally stored? Why? What is the address of the last sector of the file if it is stored in the ideal manner? [10]

(c) Just after storing the file, there is a request for reading the entire file. What is the time required for the above operation. [10]

(d) Just after the previous operation, the disk controller gets the following requests 200, 5000, 4200, 5,7200, 4000, 2200, 800, 6500 (Each number represents the cylinder number). What is the total distance travelled by the disk arm to satisfy the above requests if the disk scheduling algorithm was
A. FCFS
B. The Elevator algorithm. [10]

B+ Tree

Problem 2.

Assume that the tree is initially empty and values are inserted in ascending order. Each node can hold up to 4 pointers and 3 keys.

(a) Construct the B+ Tree that would result from successively inserting the data entries with keys A, C, E, G, I, B, D, F, H, J. [10]

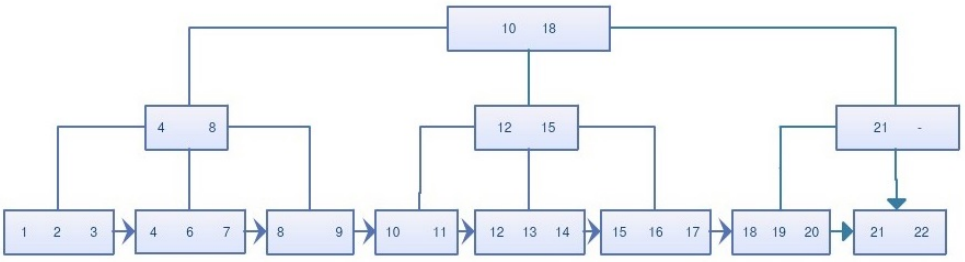
(b) Show the B+ Tree that would result from successively deleting the data entries with keys I, G and H from the original tree. [10]

Show the form of the B+ Tree after each operation.

Perform node split if required and mention/draw each split operation.

Problem 3.

Figure shows a B+ Tree. Show the modified tree that would result from inserting a data entry with key 5 into this tree. Each node can hold up to 3 keys. [10]



Show the form of the B+ Tree after each operation

Perform node split if required and mention/draw each split operation.

Problem 4.

Assume that you have just built a dense B+ Tree index on a heap file containing 30,000 records. The key field for this B+ Tree index is a 45-byte string, and it is a candidate key. Pointers (Le., record ids and page ids) are (at most) 15-byte values. The size of one disk page is 1500 bytes. The index was built in a bottom-up fashion using the bulk-loading algorithm, and the nodes at each level were filled up as much as possible.

(a) How many levels does the resulting tree have? [5]

(b) For each level of the tree, how many nodes are at that level? [5]

(c) How many levels would the resulting tree have if key compression is used and it reduces the average size of each key in an entry to 10 bytes? [5]

(d) How many levels would the resulting tree have without key compression but with all pages 70 percent full? [5]

Dynamic Hashing

Problem 5.

Show step by step approach of inserting the values: 7, 8, 10, 11, 15, 16, 17, 19, 20, 33, 53, 58 in a dynamic (extensible) hashing structure for the hash function h(x) = x mod 6. Each bucket can contain 3 records. [6]